(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 037 131 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
20.09.2000 Bulletin 2000/38

(51) Int Cl.7: **G06F 1/00**

(21) Application number: **00302122.7**

(22) Date of filing: **15.03.2000**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **15.03.1999 JP 6915199**
24.06.1999 JP 17818899

(71) Applicant: **SONY CORPORATION**
Tokyo 141 (JP)

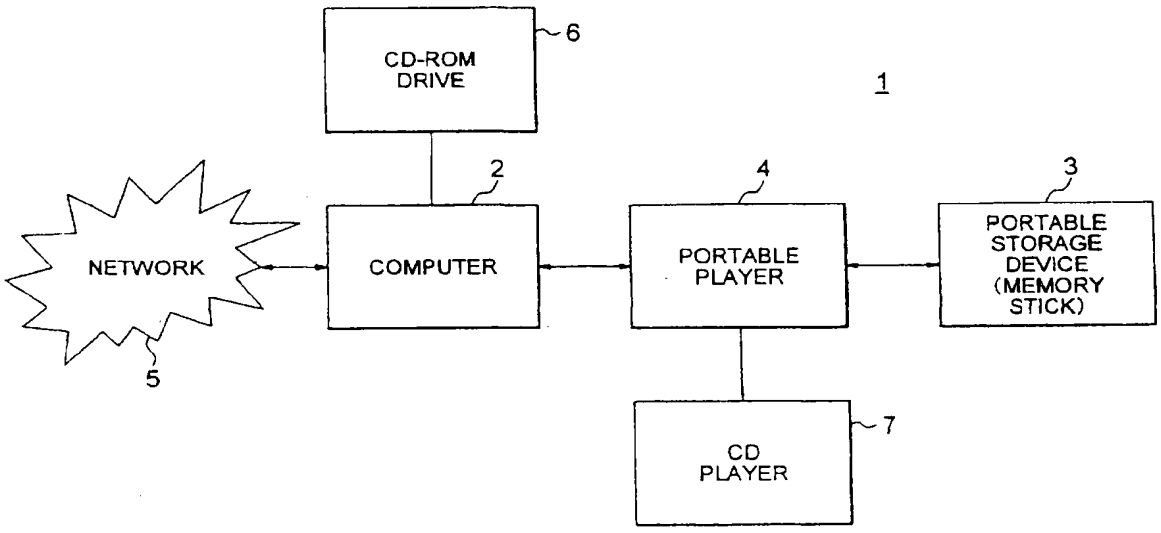
(72) Inventors:
- **Oishi, Tateo, Sony Corporation**
Tokyo (JP)
- **Asano, Tomoyuki, Sony Corporation**
Tokyo (JP)
- **Kihara, Nobuyuki, Sony Corporation**
Tokyo (JP)
- **Yokota, Teppei, Sony Corporation**
Tokyo (JP)

(74) Representative: **Turner, James Arthur et al**
D. Young & Co.,
21 New Fetter Lane
London EC4A 1DA (GB)

(54) Data processing system and method for mutual identification between apparatuses

(57) A storage unit in a first mutual identification unit stores master key data and a second storage unit in a second mutual identification unit stores identification key data. A random number generating unit generates a number that is used to select a master key and a corresponding identification key at the mutual identification units. The first mutual identification unit generates an estimate of the selected identification key using the selected master key and uses this estimate as a common key when performing mutual identification with the second mutual identification unit.

FIG. 1



EP 1 037 131 A2

Printed by Jouve, 75001 PARIS (FR)

3NSDOCID: <EP. ... 1037131A2_I. >

Description

[0001] The present invention relates to a data processing system and method for performing mutual identification between data processing apparatuses.

[0002] To prevent illicit use of audio and other data, namely, to protect the copyright(s) of such data, a first data processing apparatus may be restricted from outputting any such data to a second data processing apparatus unless a process of mutual identification between the apparatuses is performed where each apparatus is recognized as being a legitimate party.

[0003] There are a number of systems for processing such a mutual identification. One example is the common key system.

[0004] In the common key system, the first and second data processing apparatuses share a single common key. For example, one notifies the other of a random number it has generated, the two perform operations using the random number and common key, and each outputs a processing result to the other. The data processing apparatuses each compare their own processing results with the results inputted from the other, and thus, recognize the other as being a legitimate party when the results match.

[0005] In such a common key system, it is necessary to keep the common key secret from persons other than the legitimate parties. If the common key were acquired by an illegitimate party, the illegitimate party would be falsely recognized as a legitimate party under the system thus enabling illicit use of data.

[0006] Various respective aspects and features of the invention are defined in the appended claims.

[0007] To solve or at least alleviate the above problem, in accordance with the present invention, mutual identification is performed between a first data processing apparatus and a second data processing apparatus, wherein said first data processing apparatus comprises a first storage means for storing a plurality of different first key data and a first mutual identification processing means for selecting one first key data in the plurality of first key data and using the selected first key data for mutual identification with said second data processing apparatus; and said second data processing apparatus comprises a second storage means for storing a plurality of different second key data and a second mutual identification processing means for selecting a second key data corresponding to the first key data selected by the first mutual identification processing means in the plurality of second key data and using the second key data for mutual identification with said first data processing apparatus.

[0008] Further, according to a preferred embodiment of the present invention, at least one of the first and second data processing apparatuses generates a random number and notifies the generated random number to the other. Correspondingly, the first mutual identification processing means selects the first key data based on the random number and the second mutual identification processing means selects the second key data based on the random number.

[0009] According to the preferred embodiment of the invention, said first data processing apparatus further comprises a key data calculating means for calculating said second key data selected by said second mutual identification processing means from the selected first key data and said first mutual identification processing means performs the mutual identification processing with the second mutual identification processing means using the calculated second key data as a common key.

[0010] In addition, according to the preferred embodiment, said second mutual identification processing means of said second data processing apparatus performs a one-way Hash function operation using the random number input from the first data processing apparatus and the selected second key data as arguments to calculate a first processing result and outputs the first processing result to the first data processing apparatus; said first data processing apparatus further has a random number generating means for generating said random number and outputting it to said second mutual identification processing means; and said first mutual identification processing means performs said one-way Hash function operation using the random number generated by the random number generating means and the calculated second key data as arguments to produce a second processing result and identifies the second data processing apparatus as a legitimate party when the first processing result input from the second data processing apparatus matches the second processing result.

[0011] Further, according to the preferred embodiment, said first mutual identification processing means of said first data processing apparatus performs a one-way Hash function operation using the random number input from the second data processing apparatus and the calculated second key data as arguments to calculate a third processing result and outputs the third processing result to the second data processing apparatus; said second data processing apparatus further has a random number generating means for generating said random number and outputting it to said first mutual identification processing means; and said second mutual identification processing means performs said one-way Hash function operation using the random number generated by the random number generating means of the second data processing apparatus and the selected second key data as arguments to produce a fourth processing result and identifies the first data processing apparatus as a legitimate party when the third processing result input from the first data processing apparatus matches the fourth processing result.

[0012] Additionally, according to the preferred embodiment of the invention, said first data processing apparatus and said second data processing apparatus receive, as input, key selection data, wherein said first mu-

NSDOCID: <EP_____1037131A2_I_>

tual identification processing means selects one first key data in the plurality of first key data based on the key selection data, and said second mutual identification processing means selects one second key data in the plurality of second key data based on the key selection data.

[0013] Finally, the data processing method of the present invention is a method for performing mutual identification between a first data processing apparatus and a second data processing apparatus, comprising, at said first data processing apparatus, selecting one first key data in a plurality of first key data and using the selected first key data for mutual identification with said second data processing apparatus and, in said second data processing apparatus, selecting a second key data corresponding to the selected first key data in the plurality of second key data and using the selected second key data for mutual identification with said first data processing apparatus.

[0014] The invention accordingly comprises the several steps and the relation of one or more of such steps with respect to each of the others, and the apparatus embodying features of construction, combination(s) of elements and arrangement of parts that are adapted to effect such steps, all as exemplified in the following detailed disclosure, and the scope of the invention will be indicated in the claims.

[0015] The invention will now be described by way of example with reference to the accompanying drawings, throughout which like parts are referred to by like references, and in which:

Figure 1 illustrates the overall system configuration of the audio system constructed in accordance with the present invention;
Figure 2 depicts the internal structure of the portable storage device and portable player shown in Figure 1;
Figure 3 depicts data stored in a storage unit of the portable storage device shown in Figure 2;
Figure 4 depicts data stored in a flash memory of the portable device shown in Figure 2;
Figure 5 depicts a data structure of a reproduction management file PBLIST.MSF that is a sub directory stored in the portable storage device shown in Figure 2;
Figure 6 depicts the data structure of an ATRAC3 data file divided into blocks of a predetermined unit length, and including an attribute header;
Figure 7 depicts the overall data structure of a reproduction management file PBLIST;
Figure 8 depicts the detailed data structure of the reproduction management file PBLIST including: a header portion, a main data portion, and an additional information data portion;
Figure 9 depicts the structure of the data stored in the storage unit of the portable player shown in Figure 2;
Figure 10 depicts the detailed data structure of an ATRAC3 data file;
Figure 11 depicts the data structure of an upper portion of the attribute header of an ATRAC3 data file;
Figure 12 depicts the data structure of a middle portion of the attribute header of an ATRAC3 data file;
Figure 13 is a correlation table for correlating record modes, record time, and other information;
Figure 14 is a table showing copy control states;
Figure 15 depicts the data structure of a lower portion of the attribute header of an ATRAC3 data file;
Figure 16 depicts the data structure of a header of a data block of an ATRAC3 data file;
Figure 17 depicts the data stored in the storage unit of the portable player shown in Figure 2;
Figure 18 explains the CBC mode of encryption in the encrypting/decrypting unit of the portable player shown in Figure 2;
Figure 19 explains the CBC mode of decryption in the encrypting/decrypting unit of the portable player shown in Figure 2;
Figure 20 is a flow chart for explaining a write operation from the portable player to the portable storage device shown in Figure 2;
Figure 21 depicts the selection of the identification key data IKj by the mutual identification unit shown in Figure 2;
Figure 22 explains a mutual identification process between the portable storage device and the portable player shown in Figure 2;
Figure 23 explains the creation of the session key data Sek;
Figure 24 explains the write operation of audio data from the portable player to the portable storage device shown in Figure 2;
Figure 25 is a flow chart explaining a read operation from the portable storage device to the portable player shown in Figure 2;
Figure 26 explains the read operation of audio data from the portable storage device to the portable player shown in Figure 2;
Figure 27 explains the divisional editing of the track data file by the editing module of the portable player;
Figure 28 depicts the data in the cluster CL(2) of track(1) after the divisional editing shown in Figure 27;
Figure 29 depicts the data in the cluster CL(0) of track(2) after the divisional editing shown in Figure 27;
Figure 30 is a flow chart for creating the track key data and the part key data of a new track data file at the divisional editing step by the editing module of the portable player shown in Figure 2;
Figure 31 explains the coupled editing of track data files by the editing module of the portable player shown in Figure 2; and
Figure 32 is a flow chart for creating the part key data of parts (1) and (2) of track data file (3) newly

BNSDOCID: <EP 1037131A2_I_>

created in the editing module of the portable player shown in Figure 2.

[0016] Figure 1 is a view of the system configuration of an audio system 1 constructed in accordance with the present invention. The audio system 1 has for example a computer 2, a portable storage device 3, a portable player 4, a CD-ROM drive 6, and a CD player 7. Audio system 1 corresponds to the data processing system of the present invention, portable storage device 3 corresponds to the storage device of the present invention, and portable player 4 corresponds to the data processing apparatus of the present invention.

[0017] In the present embodiment, the first key data of the present invention corresponds to the content key data CK , the second key data corresponds to the part key data PK, the third key data corresponds to the temporary key data TMK, the fourth key data corresponds to the block seed data BS, and the fifth key data corresponds to the block key data.

[0018] Figure 2 is a view of the internal configuration of portable storage device 3 and portable player 4 shown in Figure 1. In the present embodiment, the module use key data calculating means of the present invention corresponds to a key creation/key processing unit 62 shown in Figure 2, the encrypting means corresponds to an encrypting/decrypting unit 64, and the key data processing means corresponds to an editing module 44.

Computer 2

[0019] Computer 2 is connected to a network 5, receives audio data (track data) from a host computer (not shown) of a service provider who provides EMD (Electronic Music Distribution) or other services via network 5, decrypts the received audio data according to need, and outputs the same to portable player 4. Computer 2 exchanges identification, charging, and other necessary information with the host computer of the service provider when receiving content data. Computer 2 can also transfer audio data input from CD-ROM drive 6 to portable player 4.

Portable Storage Device 3

[0020] As is further shown in Figure 2, portable storage device 3 houses a built in rewritable semiconductor memory such as a flash memory 34, commercially available from Sony Corporation under the trademark Memory Stick. Portable storage device 3 also has a main control module 31, a communication interface 32, a control module 33, and a flash memory management module 35.

Control Module 33

[0021] Control module 33 is a single chip integrated circuit with a multi-layer structure used exclusively for encryption. Internal memory cells are sandwiched by dummy layers such as aluminum layers. Further, control module 33 has a narrow range of operating voltage or operating frequency and is tamper resistant so that any stored data cannot be illicitly read from the outside.

[0022] As shown in Figure 2, control module 33 contains a random number generation unit 50, a storage unit 51, a key creation/processing unit 52, a mutual identification unit 53, an encrypting/decrypting unit 54, and a control unit 55. Random number generation unit 50 generates a 64 bit (8-byte) random number upon receiving a random number generation instruction. Storage unit 51 may include an EEPROM (electrically erasable programmable read only memory) or other nonvolatile memory and stores key data and other various data required for identification.

[0023] Figure 3 depicts the data stored in storage unit 51. This stored data includes identification key data IK_0 to IK_{31}, device identification data ID_m and storage use key data Skm.

[0024] The identification key data IK_0 to IK_{31} are key data used when portable storage device 3 performs a mutual identification process with portable player 4. One identification key from among the identification key data IK_0 to IK_{31} is selected at random whenever the mutual identification process is performed. Note that the identification key data IK_0 to IK_{31} and the storage use key data Skm cannot be read from outside of portable storage device 3. The device identification data ID_m is identification data uniquely attached to each portable storage device 3 and is read out when portable storage device 3 performs the mutual identification process with portable player 4. The storage use key data Skm is used when encrypting the content key data CK and storing the same in flash memory 34 as will be mentioned later.

[0025] Key creation/processing unit 52 creates the key data by performing a MAC (message authentication code) operation and/or other various operations as defined by the ISO/IEC9797 standard. Currently, the MAC operation uses a "block cipher algorithm" defined in FIPSPUB46-2 as the DES (data encryption standard). The MAC operation is a one-way Hash function in which data having an arbitrary length is compressed to a fixed length, and a function value is determined by a secret key.

[0026] Mutual identification unit 53 performs the mutual identification process with portable player 4 before receiving audio data from portable player 4 and writing the same into flash memory 34. Mutual identification unit 53 performs the mutual identification process with portable player 4 before reading audio data from flash memory 34 and outputting the same to portable player 4. Mutual identification unit 53 also performs a MAC operation as part of the mutual identification process. The data stored in storage unit 51 is used for performing the mutual identification process.

[0027] Encrypting/decrypting unit 54 performs the en-

JSDOCID: <EP 1037131A2 I_>

cryption and decryption using one of the DES, IDEA, MISTY, or other block cipher algorithms. The mode used is an ECB (electronic code book) mode and a CBC (cipher block chaining) mode as prescribed in FIPS PUB8I "DES MODES OF OPERATION". In block encryption/decryption in accordance with the ECB and CBC modes, the designated data is encrypted/decrypted by using designated key data. Control unit 55 centrally controls processing by random number generation unit 50, storage unit 51, key creation/processing unit 52, mutual identification unit 53, and encrypting/decrypting unit 54.

Flash Memory 34

[0028] Once portable player 4 is recognized as a legitimate party by the mutual identification unit 53, the audio data input from player 4 is written into flash memory 34. Conversely, the audio data can be output from flash memory 34 to portable player 4 once portable player 4 is recognized as a legitimate party by mutual identification unit 53. Flash memory 34 has a storage capacity of 32 Mbytes.

[0029] As shown in Figure 4, flash memory 34 stores a reproduction management file 100, followed by a series of track data files 101_0, 101_1, 101_2, and 101_3. Reproduction management file 100 contains data for managing reproduction of track data files 101_0 to 101_3 Track data files 101_1, to 101_3 contain the actual track data (audio data). In the present embodiment, track data is used to mean one song's worth of audio data.

[0030] Figures 5 and 6 show how a reproduction management file is used in implementing a sample file format. ATRAC3, which is a modification of the Adaptive Transform Acoustic Coding ("ATRAC") format used in Mini-Discs™ ("MDs"), is a highly efficient encoding format for audio data. Figure 5 shows the structure of a reproduction management file. Figure 6 shows the file structure of an ATRAC3 data file. An ATRAC3 data file is composed of an attribute header and an encrypted music data area for each music program. Both the reproduction management file and the ATRAC3 attribute header have a fixed file length of 16 KB (one block).

[0031] The reproduction management file shown in Figure 5 is composed of a header, a memory card name NM-1S (for one byte code), a memory card name NM2-S (for two byte code), a program reproduction sequence table TRKTBL, and an additional information area INF-S. The attribute header (shown in Figure 6) at the beginning of the data file is composed of a header, a program name NM1 (for one byte code), a program name NM2 (for two byte code), track information TRKINF (such as track key information), part information PRTINF, and an additional track information area INF. The header contains information on the total number of parts, the track name, the size of the additional information area, and so forth.

[0032] The attribute header is followed by ATRAC3 music data. The music data is block-segmented every 16 KB, each block starting with a header. The header contains an initial value for decrypting encrypted data. Only the music data of an ATRAC3 data file is encrypted. Thus, the reproduction management file, the header, and so forth are not encrypted.

[0033] Figure 7 is a schematic diagram showing the detailed data structure of a reproduction management file. Figure 8 shows a header portion and the remaining portion of the reproduction management file of Figure 7. The reproduction management file contains a 32 byte header, a name NM1-S area (256 bytes) (for the memory card), a name NM2-S area (512 bytes), a contents key area, a MAC area, an S-YMDhms area, a reproduction sequence management table TRKTBL area (800 bytes), a memory card additional information INF-S area (14720 bytes), and a redundant header information area. The start positions for each of these areas within the reproduction management file are predefined.

[0034] As shown in Figure 8, the first 32 bytes of (0x0000) to (0x0010) are used for the header. Within the file, 16-byte areas are referred to as slots. The header is placed in the first and second slots indicated at 0x000 and 0x0010. The area denoted as "Reserved" is an undefined area. Normally, a null (0x00) is written in reserved areas. However, even if data is written to a reserved area, the data is ignored. The reserved areas are intended for use in future revisions of the file format. Option areas, when not used, are treated as reserved areas. Additionally, the reproduction management file header contains the following defined areas:

= BLKID-TL0 (4 bytes)
Meaning: BLOCKID FILE ID
Function: Identifies the top of the reproduction management file.
Value: Fixed value = "TL = 0" (for example, 0x544C2D30)

= MCode (2 bytes)
Meaning: MAKER CODE
Function: Identifies the maker and model of the recorder/player
Value: High-order 10 bits (Maker code); low-order 6 bits (model code).

= REVISION (4 bytes)
Meaning: Number of rewrite times of PBLIST
Function: Increments whenever the reproduction management file is rewritten.
Value: Starts at 0 and increments by 1.

= SY1C+L (2 bytes)
Meaning: Attribute of name (one byte code) of memory card written in NM1-S area.
Function: Represents the character code and the language code as one byte code.
Value: Character code (C): High-order one byte

)NSDOCID: <EP 1037131A2 I >

00: Non-character code, binary number
01: ASCII (American Standard Code for Information Interchange)
02: ASCII+KANA
03: Modified 8859-1
81: MS-JIS
82: KS C 5601-1989
83: GB (Great Britain) 2312-80
90: S-JIS (Japanese Industrial Standards) (for Voice)
Language code (L): Low-order one byte identifies the language based on EBU Tech 3258 standard.
00: Not set
08: German
09: English
0A: Spanish
0F: French
15: Italian
1D: Dutch
65: Korean
69: Japanese
75: Chinese

[0035] When data is not recorded, this area is all 0.

= SN2C+L (2 bytes)
Meaning: Attribute of name of memory card in NM2-S area.
Function: Represents the character code and the language coded as one byte code.
Value: Same as SN1C+L

= SINFSIZE (2 bytes)
Meaning: Total size of additional information of memory card in INF-S area.
Function: Represents the data size as an increment of 16 bytes. When data is not recorded, this area is all 0.
Value: Size: 0x0001 to 0x39C (924)

= T-TRK (2 bytes)
Meaning: TOTAL TRACK NUMBER
Function: Represents the number of total tracks.
Value: 1 to 0x0190 (Max. 400 tracks)

[0036] When data is recorded, this area is all 0.

= VerNo (2 bytes)
Meaning: Format version number
Function: Represents the major version number (high order one byte) and the minor version number (low order one byte).
Value: 0x0100 (Ver 1.0)
0x0203 (Ver 2.3)

[0037] Next, areas preceded by the header are described.

= NM1-S
Meaning: Name of memory card (as one byte code)
Function: Represents the name of the memory card as one byte code (max. 256). At the end of this area, an end code (0x00) is written. The size is calculated from the end code. When data is not recorded, null (0x00) is recorded from the beginning (0x0020) of this area for at least one byte.
Value: Various character code

= NM2-S
Meaning: Name of memory card (as two byte code)
Function: Represents the name of the memory card as two byte code (max. 512). At the end of this area, an end code (0x00) is written. The size is calculated from the end code. When data is not recorded, null (0x00) is recorded from the beginning (0x0120) of this area for at least two bytes.
Value: Various character code

= CONTENTS KEY
Meaning: Value for music program. Protected with MG(M) and stored. Same as CONTENTS KEY.
Function: Used as a key necessary for calculating MAC of S-YMDhms.
Value: 0 to 0xFFFFFFFFFFFFFFFF

= MAC
Meaning: Forged copyright information check value
Function: Represents the value generated with S-YMDhms and CONTENTS KEY.
Value: 0 to 0xFFFFFFFFFFFFFFFF

= S-YMDhms (4 bytes) (optional)
Meaning: Year, month, day, hour, minute, and second recorded by the recorder/player with a reliable clock.
Function: Identifies the last recorded date and time. In this case of EMD, this area is mandatory.
Value:

bits 25 to 31: Year 0 to 99 (1980 to 2079)
bits 21 to 24: Month 0 to 12
bits 16 to 24: Day 0 to 31
bits 11 to 15: Hour 0 to 23
bits 05 to 10: Minute 0 to 59
bits 00 to 04: Second 0 to 29 (two second interval)

= TRK-nnn
Meaning: SQN (sequence) number of ATRAC3 data file reproduced.
Function: Represents FN0 of TRKINF.
Value: 1 to 400 (0x190)

[0038] When there is no track, this area is all 0.

= INF-S

EDOCID: <EP_____1037131A2_I_>

Meaning: Additional information of memory card (for example, information with respect to photos, songs, guides, etc.)
Function: Represents variable length additional information with a header. A plurality of types of additional information may be used. Each of the types of additional information has an ID and a data size. Each additional information area including a header is composed of at least 16 bytes and a multiple of 4 bytes. For details, see the following section.
Value: Refer to the section of "Data Structure of Additional Information".

[0039] In the last slot of the reproduction management file, copies of BLKID-TL0. MCode, and REVISION from the header are redundantly written.

[0040] If a memory card is accidentally detached or the power of the recorder/player turned off while data is being recorded into the card, a termination error should be detected. As described above, a REVISION area is placed at the beginning and end of each block. Whenever data is rewritten, the value of the REVISION area is incremented. If a termination error occurs in the middle of writing a block, the value of the REVISION area at the beginning of the block will not match the value of the REVISION area at the end of the block. This discrepancy between the two REVISION areas allows termination errors to be determined with a high probability. When such an abnormal termination is detected; an alarm, such as an error message, is generated.

[0041] In addition, because the fixed value BLKID-TL0 is written at the beginning of one block (16 KB) the fixed value can be used as a reference for recovering data. In other words, the fixed value allows the type of the file to be determined. Because the fixed value BLKID-TL0 is redundantly written in the header and at the end of each block, reliability is secured. Alternatively, the entire reproduction management file can be redundantly recorded.

[0042] Because the amount of data in an ATRAC3 data file is much larger than in a track information management file, ATRAC3 data files are not redundantly recorded. Instead COONUM0 and BLOCK SERIAL values are used to help recover lost ATRAC3 data (as will be described below). In addition, one ATRAC3 data file may be composed of a plurality of blocks that are dispersed. To identify blocks of the same file, CONNUM0 is used and to identify the order of the blocks BLOCK SERIAL is used. Likewise, as noted above, a maker code (MCode) is redundantly recorded at the beginning and the end of each block, so as to identify the maker of a file which has been improperly recorded.

[0043] Figure 8 also shows the structure of an additional information area. The additional information area is composed of a header comprised of the following data, and additional variable length data:

= INF
Meaning: FIELD ID
Function: Represents the beginning of the additional information (fixed value).
Value: 0x69

= ID
Meaning: Additional information key code
Function: Represents the category of the additional information.
Value: 0 to 0xFF

= SIZE
Meaning: Size of individual additional information
Function: Represents the size of each type of additional information. Although the data size is not limited, it should be at least 16 bytes and a multiple of 4 bytes. The rest of the data should be filled with null (0x00).
Value: 16 to 14784 (0x39C0)

= MCode
Meaning: MAKER CODE
Function: Identifies the maker and model of the recorder/player.
Value: High-order 10 bits (maker code), low-order 10 bits (machine code).

= C+L
Meaning: Attribute of characters in data area starting from byte 12.
Function: Represents the character code and the language code as one byte code.
Value: Same as SNC+L

= DATA
Meaning: Individual additional information
Function: Represents each type of additional information with variable length data. Real data always starts from byte 12. The length (size) of the real data should be at least 4 bytes and a multiple of 4 bytes. The rest of the data area should be filled with null (0x00).
Value: Individually defined corresponding to the contents of each type of additional information.

[0044] Next, an explanation is made of track data files 101_0 to 101_3, as shown in Figure 9. Track data file 101_0 comprises one part that includes five clusters CL(O), CL (1), CL(2), CL(3), and CL(4). The part comprising track data file 101_0 starts at the head of cluster CL(O) and ends at a sound unit SU(4) of cluster CL(4).

[0045] Note that each of the track data files 101_1 to 101_3 has basically the same configuration shown in Figure 9, but the number of parts, the number of clusters, and the number of sound units SU contained in the cluster are independently determined and may vary between track data files.

[0046] Next, the relation between music programs

BNSDOCID: <EP_____1037131A2_I_>

and ATRAC3 data files is described. One track is equivalent to one music program. In addition, one music program is composed of one ATRAC3 data (see Figure 6). The ATRAC3 data file is recorded one cluster at a time into the memory card 40. Each cluster has a capacity of 16 KB. Only one file is contained in each cluster. The minimum erasable unit of data for the flash memory 42 is one block. A block is synonymous with a cluster or a sector.

[0047] One music program (or track) is generally recorded in one part of a track data file. However, when the program is edited, the music program may be broken into a plurality of parts. The relationship between one or more parts containing a single music program is managed with part information PRTINF stored in the attribute header of each music program (see figure 6). The part size is represented with part size PRTSIZE (4 bytes) of the part information PRTINF. The first two bytes of the part size PRTSIZE represents the number of total clusters in the current part. The next two bytes represent the positions of the start sound unit (SU) and the end sound unit (SU) of the first and last clusters, respectively. By this marking of parts, the movement of music data which occurs during editing can be tracked.

[0048] SU is the minimum unit of a part compressed according to the ATRAC3 format. One SU is comprised of 1024 samples at 44.1 kHz (1024 x 16 bits x 2 channels) and can be compressed by a factor of 10. This corresponds to around 23 msec of audio. Normally, a single part contains several thousand SU. Thus, a cluster composed of 42 SU, stores about a second of audio.

[0049] Theoretically, the maximum number of parts comprising one track is 645. However, the actual number of parts usable in any given track is limited by the header, the program name, the additional data, and the size of the additional information.

[0050] Figure 10 is a diagram showing the data arrangement of an ATRAC3 data file A3Dnnnn where 1 SU is N bytes (for example, N = 384 bytes). Figure 10 also shows an attribute header (1 block) of a data file and a music data file (1 block)along with the first byte (0x0000 to 0x7FFF) of each slot of the two blocks (16 x 2 = 32 kbytes). As shown in Figure 11, the first 32 bytes of the attribute header are used as a header; 256 bytes are used as a music program area NM1 (256 bytes); and 512 bytes are used as a music program title area NM2 (512 bytes). The header of the ATRAC3 data file contains the following areas:

= BLKID-HD0 (4 bytes)
Meaning: BLOCKID FIELD ID
Function: Identifies the top of an ATRAC3 data file.
Value: Fixed value = "HD = 0" (For example, 0x48442D30)

= MCode (2 bytes)
Meaning: MAKER CODE
Function: Identifies the maker and model of the recorder/player
Value: High-order 10 bits (maker code); low- order 6 bits (machine code)

= BLOCK SERIAL (4 bytes)
Meaning: Track serial number
Function: Starts from 0 and increments by 1. Even if a music program is edited, this value does not vary.
Value: 0 to 0XFFFFFFFF.

= N1C+L (2 bytes)
Meaning: Represents the attribute of data (NM1) of a track (music program title).
Function: Represent the character code and language code of NM1 as one byte code.
Value: Same as SN1C+L

= N2C+L (2 bytes)
Meaning: Represents the attribute of data (NM2) of a track (music program title).
Function: Represent the character code and language code of NM1 as one byte code.
Value: Same as SN1C+L

= INFSIZE (2 bytes)
Meaning: Total size of additional information of current track.
Function: Represents the data size as a multiple of 16 bytes. When data is not recorded, this area should be all 0.
Value: 0x0000 to 0x3C6 (966)

= T-PRT (2 bytes)
Meaning: Number of total bytes
Function: Represents the number of parts that composes the current track. Normally, the value of T-PRT is 1.
Value: 1 to 285 (645 dec).

= T-SU (4 bytes)
Meaning: Number of total SU.
Function: Represents the total number of SU in one track that is equivalent to the program performance duration.
Value: 0x01 to 0x001FFFFF

= INX (2 bytes) (Option)
Meaning: Relative position of INDEX
Function: Used as a pointer that represents the top of a representative portion of a music program. The value of INX is designated with a value of which the number of SU is divided by 4 as the current position of the program. This value of INX is equivalent to 4 times larger than the number of SU (around 93 msec).
Value: 0 to 0xFFFF (max, around 6084 sec)

NSDOCID: <EP_____1037131A2 I_>

= XT (2 bytes) (Option)
Meaning: Reproduction duration of INDEX
Function: Designates the reproduction duration designated by INX-nnn with a value of which the number of SU is divided by 4. The value of INDEX is equivalent to four times larger than the normal SU (around 93 msec).
Value: 0x0000 (no setting), 0x01 to 0xFFFE (up to 6084 sec); 0xFFFF (up to end of music program)

Next, the music program title areas NM1 and NM2 are described.
= NM1
Means: Character string of music program title
Function: Represents a music program title as one byte code (up to 256 characters) (variable length). The title area should be completed with an end code (0x00). The size should be calculated from the end code. When data is not recorded, null (0x00) should be recorded from the beginning (0x0020) of the area for at least one byte.
Value: Various character codes

= NM2
Means: Character string of music program title
Function: Represents a music program title as two byte code (up to 512 characters) (variable length). The title area should be completed with an end code (0x00). The size should be calculated from the end code. When data is not recorded, null (0x100) should be recorded from the beginning (0x0120) of the area for at least two bytes.
Value: Various character codes

[0051] Data of 80 bytes starting from the fixed position (0x320) of the attribute header is referred to as track information area TRKINF. This area is mainly used to totally manage the security information and copy control information of the particular track. Figure 12 shows a part of TRKINF. The TRKINF area contains the following areas.

= CONTENTS KEY (8 bytes)
Meaning: Value for each music program. The value of CONTENTS KEY is protected in the security block of the memory card and then stored.
Function: Used as a key for reproducing a music program. It is used to calculate the value of MAC.
Value: 0 to 0xFFFFFFFFFFFFFFFF

= MAC (8 bytes)
Meaning: Forged copyright information check value
Function: Represents the value generated with a plurality of values of TRKINF including contents cumulation numbers and a secret sequence number. The secret sequence number is a sequence number recorded in the secret area of the memory card. A non-copyright protection type recorder cannot read data from the secret area of the memory card. On the other hand, a copyright protection type recorder and a computer that operates with a program that can read data from a memory card can access the secret area.

= A (1 byte)
Meaning: Attribute of part.
Function: Represents the information of such as compression mode of a part.
Value: see discussion hereinafter (see Figures. 12 and 13).

[0052] Next, the value of area A is described. In the following description, monaural mode (N = 0 or 1) is defined as a special joint mode of which bit 7 = 1, sub signal = 0, and main signal = (L+R). A player without copyright protection capability may ignore information bits 2 and 1.
[0053] Bit 0 of the area A states whether emphasis is on or off. Bit 1 indicates skip reproduction or normal reproduction. Bit 2 designates the data type such as audio data, FAX data, or the like. Bit 3 is undefined. Mode information for ATRAC3 is represented by the combination of bits 4, 5, and 6, as shown in Figure 13. In other words, N indicates mode and is represented by 3 bits. In Figure 13, for the five types of modes listed (monaural (N = 0 or 1), LP (N = 2), SP (N = 4), EX (N = 5), and HQ (N = 7)), record duration (64 MB memory card only), data transmission rate, and the number of SU per block are provided. The number of bytes in each SU depends on the defined mode. In the monaural mode 1 SU is 136 bytes. In the LP mode 1 SU is 192 bytes. In the SP mode 1 SU is 304 bytes. In the EX mode 1 SU is 384 bytes. In the HQ mode 1 SU is 512 bytes. Bit 7 of area A represents ATRAC3 type modes (0: Dual, 1: Joint).
[0054] As an example, a 64 MB memory card used in the SP mode is described. A 64 MB memory card has 3968 blocks. In the SP mode, since 1 SU is 304 bytes, a block is comprised of 53 SUs. Hence, 1 SU is equivalent to (1024/44100) seconds. Thus, a 64 MB memory card stores (1024/44100) x 53 x (3968 - 10) = 4863 seconds = 81 minutes. The transmission rate is (44100/1024) x 304 x 8 = 104737 bps.
[0055] Referring back to Figure 12, the remainder of the areas of TRKINF will be described.

= LT (one byte)
Meaning: Reproduction restriction flag (bits 7 and 6) and security partition (bits 5 to 0).
Function: Represents a restriction of the current track.
Value: bit 7: 0 = no restriction, 1 = restriction
bit 6: 0 = not expired, 1 = expired
bits 5 to 0: security partition (reproduction prohibited other than 0)

= FNo (2 bytes)
Meaning: File number.

ENSDOCID: <EP 1037131A2 I >

Function: Represents the initially recorded track number that designates the position of the MAC calculation value recorded in the secret area of the memory card.
Value: 1 to 0x190 (400)

= MG(D) SERIAL-nnn (16 bytes)
Meaning: Represents the serial number of the security block (security IC 20) of the recorder/player.
Function: Unique value for each recorder/player
Value: 0 to 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

= CONNUM (4 bytes)
Meaning: Contents cumulation number
Function: Represents a unique value cumulated for each music program. The value is managed by the security block of the recorder/player. The upper limit of the value is 232 that is 4,200,000,000. Used to identify a recorded program.
Value: 0 to 0xFFFFFFFF

= YMDhms-S (4 bytes) (Option)
Meaning: Reproduction start date and time of track with reproduction restriction
Function: Represents the date and time at which data reproduction is permitted with EMD.
Value: Same as the notation of date and time of other areas

= YMDhms-E (4 bytes) (Option)
Meaning: Reproduction end date and time of track with reproduction restriction
Function: Represents the date and time at which data reproduction is expired with EMD.
Value: Same as the notation of date and time of other areas

= MT (1 byte) (Option)
Meaning: Maximum value of number of permitted reproduction times
Function: Represents the maximum number of reproduction times designated by EMD.
Value: 1 to 0xFF. When not used, the value of the area MT is 00.

= CT (1 byte) (Option)
Meaning: Number of reproduction times
Function: Represents the number of reproduction times in the number of permitted reproduction times. Whenever data is reproduced, the value of the area CT is decremented.
Value: 0x00 to 0xFF. When not used, the value of the area CT is 0x00. When bit 7 of the area LT is 1 and the value of the area CT is 00, data is prohibited from being reproduced.

= CC (1 byte)
Meaning: COPY CONTROL
Function: Controls the copy operation.
Value: (see figure 14) bits 6 and 7 represent copy control information, bits 4 and 5 represent copy control information of a high speed digital copy operation, bits 2 and 3 represent a security block authentication level, bits 0 and 1 are undefined.

[0056] Example of CC:

(bits 7 and 6)
11: Unlimited copy operation permitted
01: copy prohibited
00: one time copy operation permitted
(bits 3 and 2)
00: analog/digital input recording

[0057] MG authentication level is 0. When digital record operation using data from a CD is performed, (bits 7 and 6): 00 and (bits 3 and 2): 00.

= CN (1 byte) (Option)
Meaning: Number of permitted copy times in high speed serial copy management system
Function: Extends the copy permission with the number of copy times, not limited to one time copy permission and copy free permission. Valid only in first copy generation. The value of the area CN is decremented whenever the copy operation is performed.
Value:

00: Copy prohibited
01 to 0xFE: Number of times
0xFF: Unlimited copy times

[0058] Referring once again to Figure 10, the track information area TRKINF is followed by a 24-byte part management information area (PRTINF) starting at 0x0370. When a track is composed of a plurality of parts, the addresses of the individual parts are successively arranged in PRTINF. Figure 15 shows a portion of the PRTINF area. Next, the PRTINF area is described in order of arrangement.

= PRTSIZE (4 bytes)
Meaning: Part size
Function: Represents the size of a part. Cluster: 2 bytes (highest position), start SU: 1 byte (upper), end SU: 1 byte (lowest position).
Value:

cluster: 1 to 0x1F40 (8000)
start SU: 0 to 0xA0 (160)
end SU: 0 to 0xA0 (16) (Note that SU starts from 0.)

= PRTKEY (8 bytes)

\SDOCID: <EP_____1037131A2_I_>

Meaning: Part encrypting value
Function: Encrypts a part. Initial value = 0. Note that edit rules should be applied.
Value: 0 to 0xFFFFFFFFFFFFFFFF

= CONNUM0 (4 bytes)
Meaning: Initially generated contents cumulation number key
Function: Uniquely designates an ID of contents.
Value: Same value as the value of the contents cumulation number initial value key

[0059] As is next shown in Figure 10, the attribute header of an ATRAC3 data file contains an additional information INF area. The additional information is the same as the additional information INF-S area (see Figures 7 and 8) of the reproduction management file except that the start position is not fixed. The last byte position (a multiple of four bytes) at the end of one or a plurality of parts is followed by the additional information INF area.

= INF
Meaning: Additional information with respect to track
Function: Represents variable length additional information with a header. A plurality of different types of additional information may be arranged. Each of additional information areas has an ID and a data size. Each additional information area is composed of at least 16 bytes and a multiple of 4 bytes.
Value: Same as additional information INF-S of reproduction management file

[0060] The above-described attribute header is followed by a plurality of data blocks. To each data block a header is added. Next, each block of the added header as shown in Figure 16 is described.

= BLKID-A3D (4 bytes)
Meaning: BLOCKID FILE ID
Function: Identifies the top of ATRAC3 data.
Value: Fixed value = "A3D" (for example, 0x41334420)

= MCode (2 bytes)
Meaning: MAKER CODE
Function: Identifies the maker and model of the recorder/player
Value: High-order 10 bits (maker code); low- order 6 bits (model code)

= CONNUMO (4 bytes)
Meaning: Cumulated number of initially created contents
Function: Designates a unique ID for contents. Even if the contents are edited, the value of the area CONNUMO is not changed.
Value: Same as the contents cumulation number initial key

= BLOCK SERIAL (4 bytes)
Meaning: Serial number assigned to each track
Function: Starts from 0 and increments by 1. Even if the contents are edited, the value of the area BLOCK SERIAL is not changed.
Value: 0 to 0xFFFFFFFF

= BLOCK-SEED (8 bytes)
Meaning: Key for encrypting one block
Function: The beginning of the block is a random number generated by the security block of the recorder/player. The random number is followed by a value incremented by 1. When the value of the area BLOCK-SEED is lost, since sound is not generated for around one second equivalent to one block, the same data is written to the header and the end of the block. Even if the contents are edited, the value of the area BLOCK-SEED is not changed.
Value: Initially 8-bit random number

= INITIALIZATION VECTOR (8 bytes)
Meaning: Value necessary for encrypting/decrypting ATRAC3 data
Function: Represents an initial value necessary for encrypting and decrypting ATRAC3 data for each block. A block starts from 0. The next block starts from the last encrypted 8-bit value at the last SU. When a block is divided, the last eight bytes just before the start SU is used. Even if the contents are edited, the value of the area INITIALIZATION VECTOR is not changed.
Value: 0 to 0xFFFFFFFFFFFFFFFF

= SU-nnn
Meaning: Data of sound unit
Function: Represents data compressed from 1024 samples. The number of bytes of output data depends on the compression mode. Even if the contents are edited, the value of the area SU-nnn is not changed. For example, in the SP mode, N = 384 bytes.
Value: Data value of ATRAC3

[0061] In Figure 10, since N = 384, 42 SUs are written to one block. The first two slots (4 bytes) of the block are used as a header. In the last slot (two bytes), BLKID-A3D, MCode, CONNUM0, and BLOCK SERIAL are redundantly written. Thus, M bytes of the remaining area of one block is (16,384 - 384 x 42 - 16 x 3 = 208) bytes. As described above, the eight-byte area BLOCK SEED is also redundantly recorded.

[0062] Further, the sound units SU(0) ~ (101) are each comprised of an 8-byte cryptogram C_i created by encryption in units of 64-bit (8-byte) cipher blocks in the CBC (cipher block chaining) mode in the encrypting/de-

3NSDOCID: <EP 1037131A2 I >

crypting unit 64 shown in Figure 2. In the present embodiment, the number of bytes (for example 160 bytes) of the sound unit SU is made a whole multiple of the number of bytes (for example 8 bytes) of a cipher block, that is, the unit of encryption. Namely, one sound unit SU is comprised of, for example, 20 cryptograms C_i. Each cryptogram C_i is located within a sound unit SU, and a cryptogram Ci never straddles a plurality of sound units SUs.

[0063] The audio data stored in flash memory 34 is compressed as mentioned later. The unit of compression is the sound unit SU. Accordingly, when audio data is read from the portable storage device 3 to the portable player 4, the minimum readable unit is a sound unit SU. Because of this, there are no breaks between encryption blocks and the processing load to access the data is thereby reduced when accessing encrypted audio data stored in the flash memory 34. Note that the number of sound units SU contained in each cluster may be any number within a range from 1 to 102. Further, the compression method for the audio data may be ATRAC3 or another CODEC method.

[0064] The block seed data BS is created by generating random numbers for every cluster and, as mentioned later, is used when creating block key data BK for every block in portable player 4. The block seed data BS is stored in two positions in each block as a countermeasure against errors. Further, the sound units in each of the clusters is stored at consecutive addresses in flash memory 34 by order of encryption. Thus, the encryption blocks are stored consecutively in flash memory 34 in the order of encryption.

Flash Memory Management Module 35

[0065] The flash memory management module 35 performs control for the writing of data to the flash memory 34 and for the reading of data from the flash memory 34, etc...

Portable Player 4

[0066] As shown once again in Figure 2, portable player 4 is comprised of a main control module 41, a communication interface 42, a control module 43, an edit module 44, a compression/expansion module 45, a speaker 46, a D/A converter 47, and an A/C converter 48.

Main Control Module 41

[0067] Main control module 41 centrally controls processing by the portable player 4.

Control Module 43

[0068] Control module 43 includes a random number generation unit 60, a storage unit 61, a key creation/key processing unit 62, a mutual identification unit 63, an encrypting/decrypting unit 64, and a control unit 65.

[0069] Control module 43 may be formed as a single chip integrated circuit with a multi-layer structure similar to the control module 33 used exclusively for encryption . The internal memory cells are sandwiched between dummy layers (such as aluminum layers). Further, control module 43 has a narrow range of operating voltage or operating frequency and is tamper resistant to prevent data form being illicitly read from the outside.

[0070] Random number generation unit 60 generates a 64bit (8-byte) random number upon receipt of a random number generation instruction.

[0071] Storage unit 61 stores various data required for the identification. As shown in Figure 17, storage unit 61 stores the master key data MK_0, to MK_{31} and the device identification data ID_m.

[0072] Equation (1) below shows the relationship between the master key data MK_0 to MK_{31}, identification keys IK_0 to IK_{31}, and the device identification data ID_m. In the following equation, f(a b) is a function for deriving a value from the arguments a and b.

$$IK_j = f(MK_j, ID_m) \quad (1)$$

where, j is an integer satisfying $0 \leq j \leq 31$.

[0073] The storage addresses of the identification keys IK_0 to IK_{31} in the storage unit 61 are expressed by 5 bits. They are assigned corresponding storage addresses with those for the master key data MK_0 to MK_{31} in the storage unit 51.

[0074] Key creation/key processing unit 62 creates the key data by performing various operations (e.g., the MAC operation defined in ISO/IEC9797). At this time, DES prescribed in FIPS PUB 46-2 is used as the "block cipher algorithm".

[0075] Mutual identification unit 63 performs a mutual identification process with the portable storage device 3 prior to the transfer of audio data from computer 2 to portable storage device 3. Mutual identification unit 63 also performs a mutual identification process with portable storage device 3 before the receipt of audio data from portable storage device 3. Further, mutual identification unit 63 performs the MAC operation and makes use of the data stored in the storage unit 61, during a mutual identification process. Mutual identification unit 63 performs the mutual identification process with computer 2 or the computer on network 5 before audio data is input or output to those devices.

[0076] Encrypting/decrypting unit 64 performs block encryption by selectively using the ECB mode or CBC mode described in FIPS PUB 81. Encrypting/decrypting unit 64 uses the 56-bit key k in the CBC mode to encrypt audio data (plain text) input from computer 2 or CD player 7 in units of cipher blocks consisting of 64 bits based on equation (2) below, thereby creating the encrypted

SDOCID: <EP_____1037131A2_I_>

audio data (cryptogram).

[0077] In the CBC mode, the preceding is used to encrypt the following group of data. Thus, a different cryptogram is output even when identical data is input. This makes deciphering difficult.

$$C_i = E_k (P_i \text{ XOR } C_{i-1}) \quad (2)$$

where,

i: integer of 1 or more
P_i: plain text (64 bits)
C_i: cryptogram (64 bits)
XOR: exclusive OR, and
E_k: encryption by DES system using 56-bit key data k

[0078] The operation of equation (2) will now be explained making reference to Figure 18. "IV" is the block encryption initial value (64 bits) and is stored immediately preceding the sound unit SU(0) in the cluster CL in the flash memory 34 of portable storage device 3.

[0079] ATRAC is the coding and compressing method used in MiniDisks®, and in which a 288 kbit/s 44.1 kHz sample stereo signal is encoded using band division and MDCT (modified discrete cosine transform) First, data is divided into three bands of 1/4, 1/4, and 1/2, by a band division filter, signals of bands are down-sampled and converted into the frequency domain by MDCT and the coefficient of the MDCT is scalarly quantized by adaptive bit distribution.

[0080] Encrypting/decrypting unit 64 selectively performs decryption using the ECB mode and CBC mode 15 from amongst the FIPS81 modes. Encrypting/decrypting unit 64 creates plain text by decrypting a cryptogram in units of cypher blocks using the 56-bit key k in accordance with equation (3), shown below.

$$P_i = C_{i-1} \text{ XOR } D_k (C_i) \quad (3)$$

where,

i: integer of 1 or more
P_i: plain text (64 bits)
C_i: cryptogram (64 bits)
XOR: exclusive OR, and
D_k: decryption of DES system using 56-bit key data

[0081] The operation of equation (3) will now be explained making reference to Figure 19. Note that, in Figure 19, "IV" is the block encryption initial value (64 bits) and is stored immediately preceding the sound unit SU (0) in the cluster CL in the flash memory 34 of the portable storage device 3.

[0082] Control unit 65 centrally controls the processing of random number generation unit 60, storage unit 61, key creation/processing unit 62, mutual identification unit 63, and the encrypting/decrypting unit 64.

Editing Module 44

[0083] Editing module 44 edits the track data files 101_0 to 101_3 (see figure 4) stored in flash memory 34 of portable storage device 3 to create a new track data file based on an instruction from the user. This editing can include divisional editing where one track data file is divided into two track data files, and coupled editing where two track data files are merged into one track data file. As a result of such editing, the reproduction management file 100 and the track data files 101_0 to 101_3 are rewritten as necessary.

Compression/Expansion Module 45

[0084] As part of the process to reproduce encrypted audio data, the compression/expansion module 45 expands compressed ATRAC3 audio data and outputs it to D/A converter 47. Further, module 45 compresses audio data using the ATRAC3 format when storing audio data input from CD player 7 or computer 2 in portable storage device 3.

D/A Converter 47

[0085] The D/A converter 47 converts digital audio data input from compression/expansion module 45 to analog audio data which is output to the speaker 46.

Speaker 46

[0086] The speaker 46 outputs sound according to the audio data input from the D/A converter 47.

A/D Converter 48

[0087] The A/D converter 48 converts analog audio data input from CD player 7 into digital data for output to compression/expansion module 45.

Write Operation to Portable Storage Device 3

[0088] Figure 20 is a flow chart explaining an operation for writing data from portable player 4 to portable storage device 3.

[0089] Step S1: A write request signal is sent from portable player 4 to portable storage device 3.

[0090] Step S2: The identification key data IK_j used for mutual identification between portable storage device 3 and portable player 4 is selected. The processing in this step is explained in greater detail below.

[0091] Step S3: A mutual identification process is performed between portable storage device 3 and portable player 4. The processing in this step is explained in

3NSDOCID: <EP_ _1037131A2 I_>

greater detail below.

[0092] Step S4: When portable storage device 3 and portable player 4 each recognize that the other party is legitimate in accordance with the mutual identification process of step S3, control passes to step S5. Otherwise, processing is terminated.

[0093] Step S5: A session key data Sek is created in both portable storage device 3 and portable player 4. The processing in this step will be explained in greater detail below.

[0094] Step S6: Encrypted audio data is output and written from portable player 4 to portable storage device 3 via communication interfaces 32 and 42. The processing in this step will be explained in greater detail below.

[0095] In this manner, a mutual identification process is carried out between portable storage device 3 and portable player 4, and the encrypted audio data is written from portable player 4 to portable storage device 3 only when each recognizes the other party as legitimate. By this method, illicit copying of audio data is easily avoided.

Selection of Identification Key Data IK_j (step S2 of Figure 20)

[0096] Figure 21 explains the selection of the identification key data IK_j as originally noted in step S2 of Figure 20. A 64-bit random number R_j is created by random number generation unit 60 of portable player 4 shown in Figure 2. The random number R_j is output from portable player 4 to portable storage device 3. Mutual identification unit 53 of portable storage device 3 uses the lower significant 5 bits of the 64-bit random number R_j to specify an identification key data IK_j (where j is an integer satisfying $0 \leq j \leq 31$) from the prestored identification key data IK_0 to IK_{31} stored in storage unit 51. The device identification data ID_m is similarly read from storage unit 51 of portable storage device 3 and output to portable player 4. The mutual identification unit 63 of portable player 4 uses the lower significant 5 bits of the random number R_j to specify a master key data MK_j from the prestored master key data MK_0 to MK_{31}.

[0097] Key creation/key processing unit 62 uses the specified master key data MKj and the device identification data ID_m to create the identification key data IKj based on equation (4), shown below. Note, f(a, b) is any function for deriving a value from, for example, the arguments a and b.

$$IK_j = f(MK_j, ID_m) \quad (4)$$

[0098] Once portable storage device 3 and portable player 4 have the identification key data IK_0 to IK_{31} and the master key data MK_0 to MK_{31} having the relationship shown in equation (4), the same identification key data IK_j is selected by the processing shown in Figure 21.

[0099] The selected identification key data IK_j is used as the secret key in the mutual identification process, as will be described below. Whenever the processing shown in Figure 21 is carried out, the identification key data is selected at random according to the random number R_j from among the 32 identification key data IK_j. This reduces the probability of successfully faking illicit identification to 1/32 of that of the case where only one identification key data is used, thus providing a high probability of avoiding illicit identification.

[0100] In the above embodiment, the identification key data is selected using a random number. But it is also possible to determine the identification key data based on a key designation signal input from outside of portable storage device 3 and portable player 4.

Mutual Identification Between Portable Storage Device 3 and Portable Player 4 (step S3 of Figure 20)

[0101] Figure 22 is a view for explaining the mutual identification process performed between portable storage device 3 and portable player 4. Prior to starting the mutual identification process, the selection of the identification key data IKj shown in Figure 21 has been completed and mutual identification unit 53 of portable player 4 has the selected identification key data IK_j and the device identification data ID_m. Further, mutual identification unit 63 of portable storage device 3 has the selected identification key data IK_j and the device identification data ID_m of portable storage device 3. The mutual identification proceeds as follows.

[0102] Step S10: Random number generation unit 50 of portable storage device 3 creates a 64-bit random number R_{ms} and outputs it to portable player 4.

[0103] Step S11: Random number generation unit 60 of portable player 4 creates the 64-bit random numbers R_d and S_d.

[0104] Step S12: Mutual identification unit 63 of portable player 4 uses the identification key data IK_j obtained at step S2 shown in Figure 20 and "R_d II R_{ms} II ID_m" to carry out a MAC operation based on equation (5) as shown below, to find MAC_A. Here, A II B indicates the coupling of A and B (coupling of m-bit B to end of n-bit A to make (n+m) bits).

$$MAC_A = MAC(IK_j, R_d \text{ II } R_{ms} \text{ II } ID_m) \quad (5)$$

[0105] Step S13: Portable player 4 outputs "R_d II S_d II MAC_A II j" to portable storage device 3.

[0106] Step S14: Mutual identification unit 53 of portable storage device 3 uses identification key data IK_j obtained at step S2 shown in Figure 20 and the "R_d II R_{ms} II ID_m" to carry out a MAC operation based on equation (6), as shown below, to find MAC_B.

NSDOCID: <EP_____1037131A2 I_>

$$MAC_B = MAC\ (IK_j,\ R_d\ ||\ R_{ms}\ ||\ ID_m) \qquad (6)$$

[0107] Step S15: Mutual identification unit 53 of portable storage device 3 compares the MAC_B found at step S14 and the MAC_A input at step S13. If they coincide, the portable player 4 has an adequate identification key data IK_j, so portable storage device 3 identifies it as a legitimate party.

[0108] Step S16: Mutual identification 53 of portable storage device 3 uses identification key data IK_j obtained at step S2 shown in Figure 20 and "R_{ms} || R_d" to carry out a MAC operation based on equation (7) to find MAC_C.

$$MAC_C = MAC\ (IK_j,\ R_{ms}\ ||\ R_d) \qquad (7)$$

[0109] Step S17: Random number generation unit 50 of portable storage device 3 creates the 64-bit random number S_{ms}.

[0110] Step S18: "S_{ms} || MAC_C" is output from portable storage device 3 to portable player 4.

[0111] Step S19: Mutual identification unit 63 of portable player 4 carries out the MAC operation based on equation (8) to find MAC_d.

$$MAC_d = MAC\ (IKj,\ R_{ms}\ ||\ R_d) \qquad (8)$$

[0112] Step S20: Mutual identification unit 53 of portable player 4 compares MAC_d found at step S19 and MAC_C input at step S18. If they coincide, portable storage device 3 has an adequate identification key data IK_j, so portable player 4 identifies it as a legitimate party.

[0113] In accordance with the above, mutual identification between portable storage device 3 and portable player 4 is achieved.

Creation of Session Key Data Sek (step S5 of Figure 20)

[0114] Figure 23 explains the creation of the session key data Sek. Prior to starting the creation of the session key data Sek, the selection of the identification key data IK_j shown in Figure 21 and the mutual identification process shown in Figure 22 are complete. Both portable storage device 3 and portable player 4 have the selected identification key data IK_j and the random numbers S_d and S_{ms}. The creation of session key data Sek progresses as follows.

[0115] Step S30: Mutual identification unit 63 of portable player 4 uses the selected identification key data IKj and "S_d || S_{ms}" to perform a MAC operation based on equation (9) to create the session key data Sek.

$$Sek = MAC\ (IK_j,\ S_d\ ||\ S_{ms}) \qquad (9)$$

[0116] Step S31: Mutual identification unit 53 of portable storage device 3 uses the selected identification key data IK_j and "S_d || S_{ms}" to perform a MAC operation based on equation (10) to create the session key data Sek.

$$Sek = MAC\ (IK_j,\ S_d\ ||\ S_{ms}) \qquad (10)$$

[0117] The session key data Sek created at portable storage device 3 is the same as the session key data Sek created at portable player 4 if both parties are legitimate.

Writing of Audio Data into Portable Storage Device 3 (step S6 of Figure 20)

[0118] Figure 24 explains the write processing of audio data from portable player 4 into portable storage device 3. Prior to starting the write processing, the creation processing of the session key data Sek shown in Figure 23 has been completed and portable storage device 3 and portable player 4 have the same session key data Sek. The writing of audio data into portable storage device 3 progresses as follows.

[0119] Step S40: Portable player 4 requests random number generation unit 60 to generate a random number for each track and create a corresponding content key data CK according to each of the random numbers.

[0120] Step S41: Portable player 4 encrypts content key data CK created at step S40 in encrypting/decrypting unit 64 by using the session key data Sek.

[0121] Step S42: Portable player 4 outputs content key data CK encrypted at step S41 to portable storage device 3.

[0122] Step S43: Portable storage device 3 decrypts encrypted content key data CK input at step S42 in encrypting/decrypting unit 54.

[0123] Step S44: Portable storage device 3 encrypts content key data CK decrypted at step S43 in the encrypting/decrypting unit 54 by using the storage use key data Skm read from storage unit 51.

[0124] Step S45: Portable storage device 3 outputs the encrypted CK to the portable player 4.

[0125] Step S46: Portable player 4 sets the related encrypted content key data CK in the TRKINF in track data file 101_n.

[0126] Step S47: Random number generation unit 60 generates a random number for each part of a track data file and creates part key data PK according to the random number. The created part key data PK is set in the management data PRTINF of track data file 101_n.

[0127] Step S48: The XOR of part key data PK created at step S45 and content key data CK is obtained in key creation/processing unit 62 for each part of the track data file as shown below in equation (11). The result of

3NSDOCID: <EP_____1037131A2_I_>

the processing is the generation of a temporary key data TMK. The creation of temporary key data TMK is not limited to using an XOR function. It is possible to use other functional operators, such as a simple AND operator.

$$TMK = PK \text{ XOR } CK \quad (11)$$

[0128] Step S49: Random number generation unit 60 generates a random number for each block and creates block seed data BS according to the random number. Further, portable player 4 sets the created block seed data BS into its proper position in each corresponding block.

[0129] Step S50: Key creation/key processing unit 62 uses the temporary key data TMK created at step S46 and the block seed data BS created at step S47 in equation (12) to perform a MAC operation and create block key data BK for each block.

$$BK = MAC(TMK, BS) \quad (12)$$

[0130] It is possible to perform processing other than a MAC operation by using the secret key on the input of a SHA-1 (secure Hash algorithm), RIPEMD-160, or other one-way Hash function to create block key data BK. Here, the one-way function f defines a function from which it is easy to calculate $y = f(x)$ from x, but conversely difficult to find x from y. A one-way Hash function is described in detail in the "Handbook of Applied Cryptography, CRC Press".

[0131] Step S51: Portable player 4 compresses the audio data input from computer 2 or portable player 4 in accordance with the ATRAC3 format in compression/expansion module 45. Then, encrypting/decrypting unit 64 encrypts the compressed audio data in the CBC mode by using the block key data BK created at step S50.

[0132] Step S52: Portable player 4 adds headers to the audio data encrypted at step S51 and outputs them via the communication interfaces 32 and 42 to portable storage device 3.

[0133] Step S53: Portable storage device 3 writes the encrypted audio data and header into flash memory 34.

[0134] At this point, writing of the audio data from the portable player 4 to the portable player 4 is complete. Although the above description only discusses writing track data files 101_0 to 101_3, the portable player 4 also writes the reproduction management file 100 in this manner.

Reading from Portable Storage Device 3

[0135] Figure 25 is a flow chart explaining a read operation for reading data from portable storage device 3 to portable player 4.

[0136] Step S61: A read request signal specifying a desired track data (tune) is sent from portable player 4 to portable storage device 3.

[0137] Step S2: The identification key data IK_j used when performing the mutual identification between the portable storage device 3 and the portable player 4 is selected in a manner as described above.

[0138] Step S3: Mutual identification processing is performed between portable storage device 3 and portable player 4 in a manner as described above.

[0139] Step S4: Where both portable storage device 3 and portable player 4 identify each other as legitimate, the processing proceeds. Otherwise, processing is terminated.

[0140] Step S5: Session key data Sek is created at portable storage device 3 and portable player 4 in a manner as described above.

[0141] Step S63: The encrypted audio data is read via communication interfaces 32 and 42 from portable storage device 3 to portable player 4. This processing will be explained in greater detail below.

[0142] Mutual identification is carried out between portable storage device 3 and portable player 4. Only when the two parties identify each other as legitimate can the encrypted content key data be decrypted using the proper session key data Sek. Therefore, illicit utilization of the audio data is easily avoided.

Reading of Audio Data from Portable Storage Device 3 (step S63 of Figure 25)

[0143] Figure 26 explains the reading of audio data from portable storage device 3 to portable player 4. This reading step requires the data to be written by the above described method. The writing of the track data files 101_0 to 101_3 is critical to set the content key data CK in the TRKINF, the part key data PK in the PRTINF, and the block seed data BS in each cluster CL. Because the processing of step S5 is complete, portable storage device 3 and portable player 4 have the same session key data. The reading of audio data from portable storage device 3 proceeds as follows.

[0144] Step S71: Portable storage device 3 specifies the track data file corresponding to the read request signal and outputs the audio data in sound units SUs from the cluster comprising the specified track data. Portable storage device 3 also reads out the corresponding attribute header of the audio data and outputs it to portable player 4.

[0145] Step S72: Portable player 4 picks-up the CK from the TRKINF in the input attribute header and outputs it to portable storage device 3.

[0146] Step S73: Encrypting/decrypting unit 54 of portable storage device 3 decrypts the content key data CK input at step S72 using the storage key data Skm stored in storage unit 51.

[0147] Step S74: Encrypting/decrypting unit 54 of portable storage device 3 encrypts the content key data CK decrypted at step S73 using the session key data

\SDOCID: <EP 1037131A2 I_>

Sek obtained at step S5 shown in Figure 25.

[0148] Step S75: Portable storage device 3 outputs the content key data CK encrypted at step S74 to portable player 4.

[0149] Step S76: Encrypting/decrypting unit 64 of portable player 4 decrypts the content key data CK input from the portable storage device 3 at step S73 using the session key data Sek.

[0150] Step S77: Key creation/processing unit 62 of portable player 4 obtains the XOR of the content key data CK decrypted at step S76 and the part key data PK from the PRTINF in the attribute header being input at step S71 and defines the result of the processing as the temporary key data TMK in accordance with equation (13).

$$TMK = PK \text{ XOR } CK \quad (13)$$

[0151] Step S78: Key creation/key processing unit 62 of portable player 4 uses the temporary key data TMK created at step S77 and the block seed data BS in the track data file inputted at step S71 to perform the MAC operation shown in the following equation (14) and defines the result of the processing as the block key data BK. The block key data BK is found for every cluster (block) as follows.

$$BK = MAC\ (TMK, BS) \quad (14)$$

[0152] Step S79: Portable player 4 decrypts the audio data input at step S71 in encrypting/decrypting unit 64 by using the block key data BK created at step S78.

[0153] At this point, the audio data is decrypted for every cluster (block) using the individually found block key data BK. Further, decryption is carried out in the same 8-byte blocks as used for encryption.

[0154] Step S80: Portable player 4 expands the audio data decrypted at step S79 by the ATRAC3 system in compression/expansion module 45 and converts the expanded audio data to a digital format at D/A converter 47 for output to speaker 46.

[0155] The audio data decrypted at step S78 is expanded in sound units SUs.

Divisional Editing of Track Data File

[0156] As previously mentioned, editing module 44 of portable player 4 is adapted to perform the divisional editing of dividing one track data file to create two track data files and the coupled editing of coupling two track data files to create one track data file.

[0157] First, an explanation of divisional editing is provided. Figure 27 explains the divisional editing of a track data file by editing module 44 of portable player 4. As an example, editing module 44 divides a track data file (1) shown in Figure 27(A) into a new track data file (1) shown in Figure 27(B) and a track data file (2) shown in Figure 27(C). The minimum divisible unit is the sound unit SU. In this example, the sound units SU(3) and SU (4) of cluster CL(2) are divided as shown in Figure 27(B).

[0158] After division, cluster CL(2) of track data file (1) is as shown in Figure 28, and cluster CL(0) of the newly created track data file (2) is as shown in Figure 29. Sound unit SU(4) of cluster (2) of track data file (1) before the division, becomes sound unit SU(0) of cluster CL(0) in track data file (2). Similarly, sound unit SU (5) of the cluster (2) of track data file (1) before the division, becomes sound unit SU(1) of cluster CL(0) of track data file (2).

[0159] Further, the block encryption initial value IV of cluster CL (0) of track data file (2) is set equal to the last 8 bytes of sound unit SU (3) in cluster CL(2) of the track data file (1) shown in Figs. 27(A) and 27(B). As mentioned above, in each cluster the block encryption initial value IV is arranged as the 8 bytes immediately before the first sound unit SU (0). Thus, each divided cluster contains its own encryption information, so that regardless of subsequent division, the data can be easily reproduced.

[0160] The content key data, part key data, and block key data of track data file (1) before the division are CK-1, PK-1, and BK-1. Further, the content key data, part key data, and block key data of track data file (1) after division are CK-1', PK-1', and BK-1. Also, the content key data, part key data, and block key data of track data file (2) are CK-2, PK-2, and BK-1.

[0161] Figure 30 is a flowchart explaining creation of the content key data and the part key data of new track data file (2) in editing module 44 of portable player 4. The new track data file (2) created by the division has the new content key data CK-2 separate from the track data file (1). By calculating the part key data PK-2 as shown below, the block key data BK-1 is the same as before the division. The process continues as follows.

[0162] Step S90: Editing module 44 waits until it receives a division instruction in which case control passes to step S91.

[0163] Step S91: Random number generation unit 60 generates a random number and creates the new content key data CK-2 according to the generated random number.

[0164] Step S92: Encrypting/decrypting unit 54 of portable storage device 3 encrypts the content key data CK-2 created at step S91 using the storage use key data Skm stored in storage unit 51.

[0165] Step S93: Editing module 44 writes the encrypted content key data CK-2 into the TRKINF in the corresponding track data file.

[0166] Step S94: Editing module 44 creates the part key data PK-2 of track data file (2) based on equation (15).

$$PK\text{-}2 = CK\text{-}1 \text{ XOR } PK\text{-}1 \text{ XOR } CK\text{-}2 \quad (15)$$

3NSDOCID: <EP 1037131A2_1_>

[0167] This process makes the temporary key data (from equation (11)) for track data file (2) the same as the temporary key data of track data file (1), and the cluster key data created (from equation (12)) the same as the block key BK-1 before the division. For this reason, it is not necessary to encrypt the sound units SUs in the track data file (2) again using the new block key data.

[0168] Step S95: Editing module 44 writes the part key data PK-2 created at step S94 into PRTINF in the corresponding track data file.

[0169] Thus, even when the new content key data CK-2 is the same as the content key data of the newly created track data file (2) the part key data PK-2 created based on equation (15) allows the temporary key data to be made the same as the temporary key data before the division. As a result, the block key data is also the same as before the division. Therefore, it is not necessary to encrypt the sound units SUs in track data file (2) again using the new cluster key data. Similarly, the part key data PK-1' of track data file (1) after division is determined according to the content key data CK-1, so as not to change the block key data BK-1. As a result, it is not necessary to encrypt the sound units SUs in track data file (1) after division again by using new block key data. This allows the divisional editing of a track data file while avoiding a great increase in the amount of processing. Although the above description relates only to the track data files 101_0 to 101_3, editing module 44 rewrites the reproduction management file in a corresponding manner.

[0170] Figure 31 explains coupling (or merging) of two track data files by editing module 44 of portable player 4. For example, editing module 44 couples track data file (1) shown in Figure 31(A) and track data file (2) shown in Figure 31(B) to create track data file (3) shown in Figure 31(C). By coupling, a new track data file (3) is created containing a part (1) comprised of track data file (1) prior to coupling and a part (2) comprised of track data file (2) prior to coupling.

[0171] Further, content key data CK-3 for track data file (3), part key data PK-3-1 for part (1) and part key data PK-3-2 for part (2) are newly created as will be explained in greater detail below. The newly created key data are set into TRKINF and PRTINF in track data file (3).

[0172] The clusters CL(0) and CL(4) of track data file (1) prior to coupling become the start cluster and end cluster of part (1) of track (3) after coupling. Further, the clusters CL(0) and CL(5) of track data file (2) prior to coupling become the start cluster and the end cluster of part (2) of track (3) after coupling.

[0173] Figure 32 is a flowchart explaining creation of the part key data for parts (1) and (2) of the newly created track data file (3). In the following explanation track data file (1) uses content key data CK-1, part key data PK-1, and block key data BK-1, while track data file (2) uses content key data CK-2, part key data PK-2, and block key data BK-2. Track data file (3) acquires the new content key data CK-3 by calculating the part key data of parts (1) and (2) as will be described below. The block key data BK-1 and BK-2 remains the same as before coupling. The coupling process proceeds as follows.

[0174] Step S100: Editing module 44 waits until it receives a coupling instruction in which case control passes to step S101.

[0175] Step S101: Random number generation unit 60 generates a random number and creates the content key data CK-3 accordingly.

[0176] Step S102: Encrypting/decrypting unit 54 of portable storage device 3 encrypts the content key data CK-3 created at step S101 using the storage use key data Skm stored in storage unit 51.

[0177] Step S103: Editing module 44 writes the encrypted content key data CK-3 into TRKINF of track data file (3).

[0178] Step S104: Editing module 44 creates the part key data PK-3-1 for part (1) of track data file (3) based on equation (16).

$$PK\text{-}3\text{-}1 = CK\text{-}1 \text{ XOR } PK\text{-}1 \text{ XOR } CK\text{-}3 \qquad (16)$$

Temporary key data of part (1) (from equation (11)) is therefore the same as the temporary key data of the track data file (1) prior to coupling. As a result, the block key data of part (1) (from equation (12)) is also the same as the block key data BK-1 of the track data file (1) prior to coupling. Thus, it is not necessary to encrypt the sound units SUs of part (1) again using the new block key data.

[0179] Step S105: Editing module 44 creates the part key data PK-3-2 for part (2) of track data file (3) based on equation (17).

$$PK\text{-}3\text{-}2 = CK\text{-}2 \text{ XOR } PK\text{-}2 \text{ XOR } CK\text{-}3 \qquad (17)$$

Temporary key data of part (2) is therefore the same as the temporary key data of the track data file (2) As a result, the block key data of part (2) is also the same as the block key data BK-2 of track data file (2). For this reason, it is not necessary to encrypt the sound units SUs of part (2) again using the new block key data.

[0180] Step S106: Editing module 44 writes the part key data PK-3-1 created at step S104 in PRINTF of part (1) of track data file (3).

[0181] Step S107: Editing module 44 writes the part key data PK-3-2 created at step S105 in the PRINTF of part (2) of track data file (3).

[0182] Thus, even when the new content key data CK-3 is the same as the content key data of the newly created track data file (3), the part key data PK-3-1 and PK-3-2 created based on the equations (16) and (17) allows the temporary key data for each part to be made the same as for the similar data before the coupling. As a result, the block key data of the corresponding parts is

NSDOCID: <EP_____1037131A2_I_>

also the same as BK-1 and BK-2 before the coupling. Therefore, it is not necessary to encrypt the sound units SUs in parts (1) and (2) again, using the new block key data. For this reason, the great increase in the amount of processing which normally accompanies coupled editing is avoided. Further, although the above description only relates to the track data files 101_0 to 101_3, editing module 44 correspondingly rewrites the reproduction management file 100.

[0183] The present invention is not limited to the above embodiments. For example, the above embodiments have the number of bytes (160 bytes) of the sound units SUs a whole multiple of the number of bytes (8 bytes) of the cipher block (the unit of encryption in the CBC mode). But the present invention, can be adjusted for when it is not a whole multiple by inserting padding to adjust the data length of the sound unit SU.

[0184] Further, the case was shown of first outputting the random number R_{ms} created at portable storage device 3 to portable player 4 when mutual identification processing is performed as shown in Figure 22. But it is also possible to first output a random number created at portable player 4 to portable storage device 3.

[0185] Further, the case where 32 sets of the identification key data and master key data were stored in storage units 51 and 61 was shown, but there may be any number of these sets as long as it is 2 or more.

[0186] Further, the case where the identification key data IK_0 to IK_{31} were created from the master key data MK_0 to MK_{31} in portable player 4 was given. But it is also possible to store the identification key data IK_0 to IK_{31} in portable player 4 in the same way as that for portable storage device 3 and select the identification key data according to the random number R_j.

[0187] Further, as shown in Figure 21, the case where the identification key data IK_j and the master key data MK_j are selected in portable storage device 3 and portable player 4 by using the random number R_j created at portable player 4 was exemplified. But it is also possible to use a random number created at portable storage device 3 or to use random numbers generated in both portable storage device 3 and portable player 4.

[0188] Further, the above embodiments show the case where the identification key data IK_j and the master key data MK_j were selected in portable storage device 3 and portable player 4 based on the random number R_j. But according to the present invention, it is also possible to input the 5-bit key selection instruction data to portable storage device 3 and portable player 4 from the outside and select the identification key data IK_j and the master key data MK_j corresponding to each other indicated by the related key selection instruction data at portable storage device 3 and portable player 4.

[0189] Further, an example was given of data containing audio data as the track data, but the present invention can also be applied where track data containing moving picture data, stationary image data, document data, program data, and other types of data are stored in flash memory 34.

[0190] As explained above, according to the data processing apparatus and the data processing system of the present invention and the method for the same, even in the case where the first key data is changed after encrypting the track data by using the third key data and storing the same in the storage device, the third key data is not changed, so it becomes unnecessary to decrypt and re-encrypt the track data. For this reason, the amount of processing required when the first key data is changed is greatly reduced.

[0191] It will thus be seen that the objects set forth above, among those made apparent from the preceding description, are efficiently attained and, because certain changes may be made in carrying out the above method and in the construction(s) set forth without departing from the scope of the invention, it is intended that all matter contained in the above description and shown in the accompanying drawings shall be interpreted as illustrative and not in a limiting sense.

[0192] It is also to be understood that the following claims are intended to cover all of the generic and specific features of the invention herein described and all statements of the scope of the invention which, as a matter of language, might be said to fall therein.

[0193] In so far as the embodiments of the invention described above are implemented, at least in part, using software-controlled data processing apparatus, it will be appreciated that a computer program providing such software control and a storage medium by which such a computer program is stored are envisaged as aspects of the present invention.

Claims

1. A data processing system performing mutual identification between a first data processing apparatus and a second data processing apparatus, wherein:

   said first data processing apparatus comprises:
   first storage means for storing a plurality of different first key data; and
   first mutual identification processing means for selecting a first key from the plurality of first key data and using the selected first key for mutual identification with second data processing apparatus; and
   said second data processing apparatus comprises:
   second storage means for storing a plurality of different second key data; and
   second mutual identification processing means for selecting a second key from the plurality of second key data corresponding to the first key selected by the first mutual identification processing means and using the second key for mutual identification with said first data

3NSDOCID <EP_____1037131A2_I_>

processing apparatus.

2. A data processing system as set forth in claim 1, wherein:

at least one of the first data processing apparatus and the second data processing apparatus further comprises means for generating a random number and outputting the random number;
the first mutual identification processing means selects the first key based on the random number; and
the second mutual identification processing means selects the second key based on the random number.

3. A data processing system as set forth in claim 1, wherein:

said first data processing apparatus further comprises key data calculating means for using the selected first key to calculate an estimate of the second key selected by said second mutual identification processing means; and
said first mutual identification processing means performs mutual identification with the second mutual identification processing means using the estimate of the second key as a common key.

4. A data processing system as set forth in claim 3, wherein:

said second data processing apparatus further comprises means for storing identification data and outputting said identification data to said first data processing apparatus; and
said calculating means uses the identification data inputted from said second data processing apparatus to calculate the estimate of the second key.

5. A data processing system as set forth in claim 3, wherein

said first data processing apparatus comprises a random number generating means for generating a random number and outputting the random number to said second mutual identification processing means;
said second mutual identification processing means of said second data processing apparatus performs a one-way Hash function operation using the random number inputted from the first data processing apparatus and the selected second key as arguments to calculate a first processing result and outputs the first processing result to the first data processing apparatus; and
said first mutual identification processing means performs the one-way Hash function operation using the random number generated by the random number generating means and the estimate of the second key as arguments to produce a second processing result and identifies the second data processing apparatus as a legitimate party when the first processing result inputted from the second data processing apparatus matches the second processing result.

6. A data processing system as set forth in claim 5, wherein:

said second data processing apparatus comprises a second random number generating means for generating a second random number and outputting the second random number to said first mutual identification processing means;
said first mutual identification processing means of said first data processing apparatus performs a second one-way Hash function operation using the second random number inputted from the second data processing apparatus and the estimate of the second key as arguments to calculate a third processing result and outputs the third processing result to the second data processing apparatus; and
said second mutual identification processing means performs the second one-way Hash function operation using the second random number generated by the second random number generating means of the second data processing apparatus and the selected second key as arguments to produce a fourth processing result and identifies the first data processing apparatus as a legitimate party when the third processing result inputted from the first data processing apparatus matches the fourth processing result.

7. A data processing system as set forth in claim 1, wherein

said first data processing apparatus and said second data processing apparatus receive key selection data as input;
said first mutual identification processing means selects a first key from the plurality of first key data based on the key selection data; and
said second mutual identification processing means selects a second key from the plurality of second key data based on the key selection

NSDOCID: <EP 1037131A2_I_>

data.

8. A data processing system as set forth in claim 1, wherein:

said first data processing apparatus and said second data processing apparatus output key data for decrypting data transferred between said first and second data processing apparatuses when the first mutual identification processing means and the second mutual identification processing means recognize each other as being legitimate parties.

9. A data processing system as set forth in claim 8, wherein said second data processing apparatus further comprises means for storing encoded data inputted from the first data processing apparatus.

10. A data processing method for performing mutual identification between a first data processing apparatus and a second data processing apparatus, comprising the steps of:

selecting, at said first data processing apparatus, a first key from a plurality of first key data and using the selected first key for mutual identification with said second data processing apparatus; and

selecting, at said second data processing apparatus, a second key from the plurality of second key data corresponding to the selected first key and using the selected second key for mutual identification with said first data processing apparatus.

11. A data processing method as set forth in claim 10, further comprising the steps of:

generating, by at least one of the first data processing apparatus and the second data processing apparatus, a random number and outputting the random number;

selecting at the first data processing apparatus the first key based on the random number; and

selecting at the second data processing apparatus the second key based on the random number.

12. A data processing method as set forth in claim 10, further comprising the steps of:

calculating, at the first data processing apparatus, an estimate of the second key selected by said second data processing apparatus using the selected first key; and

performing mutual identification between the first data processing apparatus and the second data processing apparatus using the estimate of the second key as a common key.

13. A data processing method as set forth in claim 12, further comprising the step of outputting identification data stored in said second data processing apparatus to said first data processing apparatus; and

wherein said calculating step includes using the identification data to calculate the estimate of the second key.

14. A data processing method as set forth in claim 12, further comprising the steps of:

generating, at said first data processing apparatus, a random number and outputting the random number to said second data processing apparatus;

performing, at said second data processing apparatus, a one-way Hash function operation using the random number inputted from the first data processing apparatus and the selected second key as arguments to calculate a first processing result and outputting the first processing result to the first data processing apparatus; and

performing, at said first data processing apparatus, the one-way Hash function operation using the random number and the estimate of the second key as arguments to produce a second processing result and identifying the second data processing apparatus as a legitimate party when the first processing result inputted from the second data processing apparatus matches the second processing result.

15. A data processing method as set forth in claim 14, further comprising the steps of:

generating, at said second data processing apparatus, a second random number and outputting the second random number to said first data processing apparatus;

performing, at said first data processing apparatus, a second one-way Hash function operation using the second random number inputted from the second data processing apparatus and the estimate of the second key as arguments to calculate a third processing result and outputs the third processing result to the second data processing apparatus; and

performing, at said second data processing apparatus, the second one-way Hash function operation using the second random number and the selected second key as arguments to produce a fourth processing result and identifying the first data processing apparatus as a legitimate party when the third processing result input from the first data processing apparatus matches he fourth processing result.

3NSDOCID: <EP 1037131A2 I >

16. A data processing method as set forth in claim 10, further comprising the steps of:

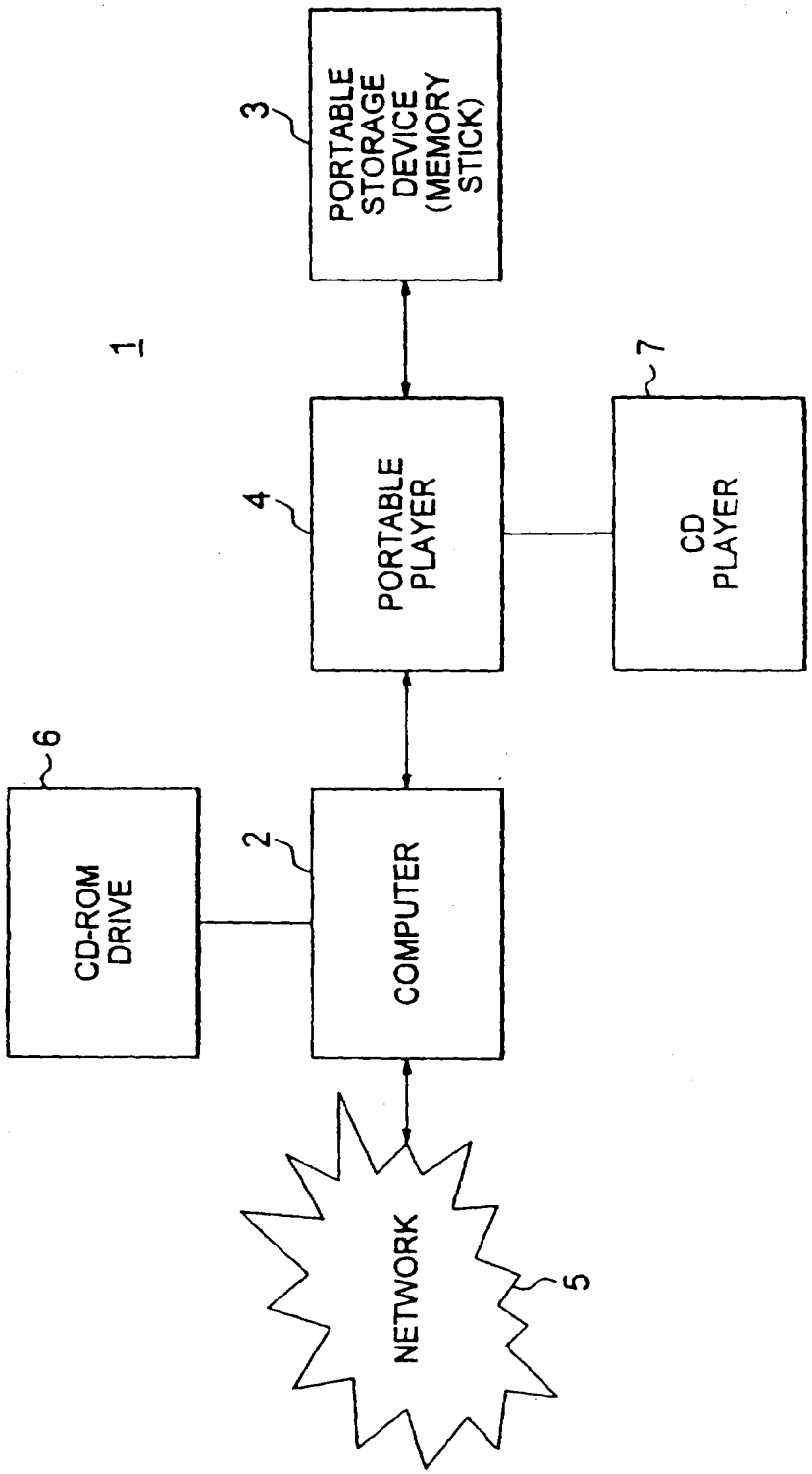
receiving, at said first data processing apparatus and said second data processing apparatus, key selection data as input;
selecting, at said first data processing apparatus, a first key from the plurality of first key data based on the key selection data; and
selecting, at said second data processing apparatus, a second key from the plurality of second key data based on the key selection data.

17. A data processing method as set forth in claim 10, further comprising the step of outputting key data for decrypting data transferred between said first and second data processing apparatuses when the first data processing apparatus and the second data processing apparatus recognize each other as being legitimate parties.

18. A data processing method as set forth in claim 17, further comprising the step of storing encoded data inputted from the first data processing apparatus to the second data processing apparatus.

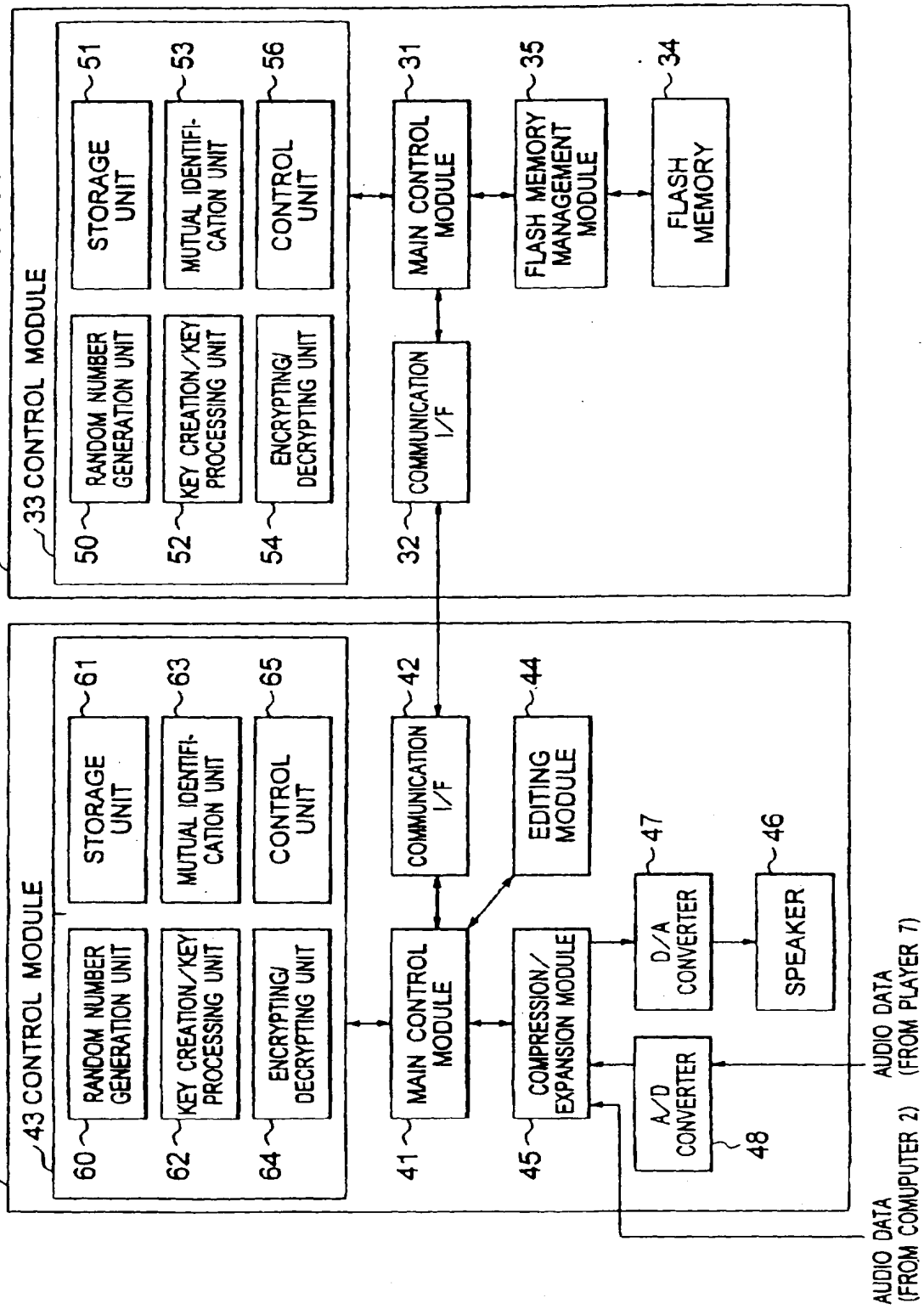
NSDOCID: <EP_____1037131A2_I_>

FIG. 1



ENSDOCID: <EP 1037131A2 I >

FIG. 2



NSDOCID: <EP 1037131A2 I >

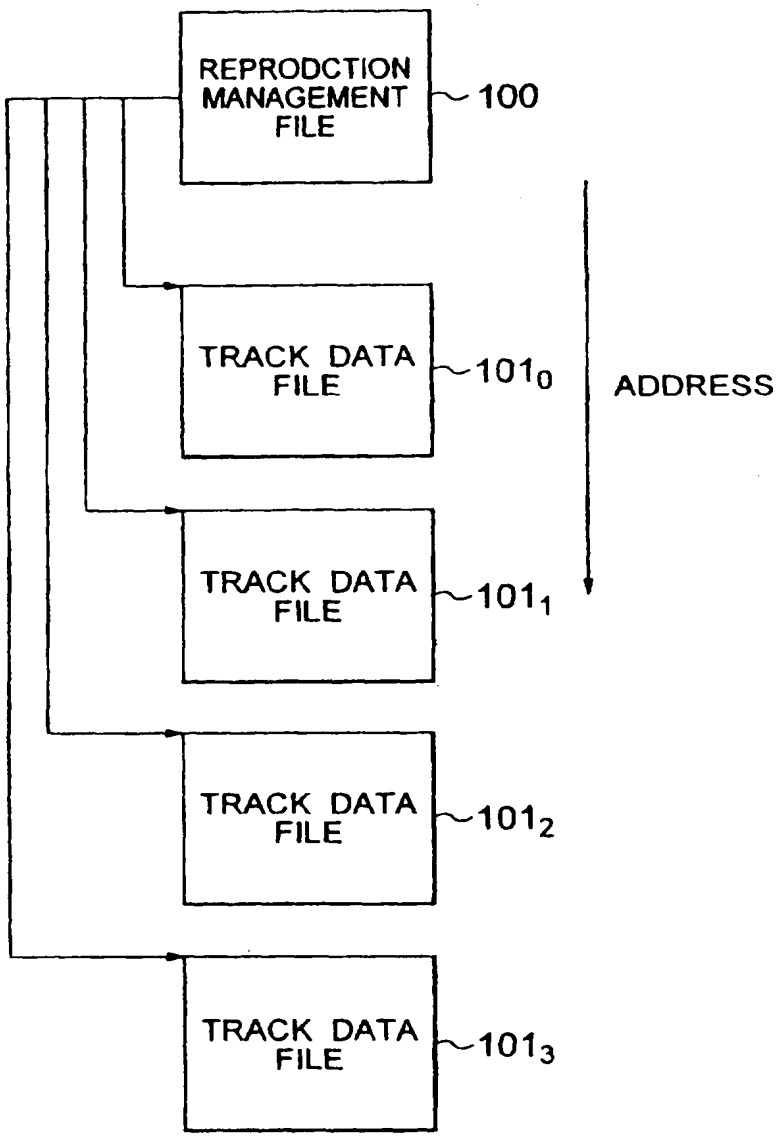
FIG. 3

DATA STOREND IN STORAGE UNIT 51 OF PORTABLE STORAGE DEVICE 3

IDENTIFICATION KEY DATA	IK_0
	IK_1
	IK_2
	IK_3
	⋮
	IK_{30}
	IK_{31}
DEVICE IDENTIFICATION DATA	ID_m
STORAGE USE KEY DATA	SK_m

3NSDOCID: <EP_____ 1037131A2 I_>

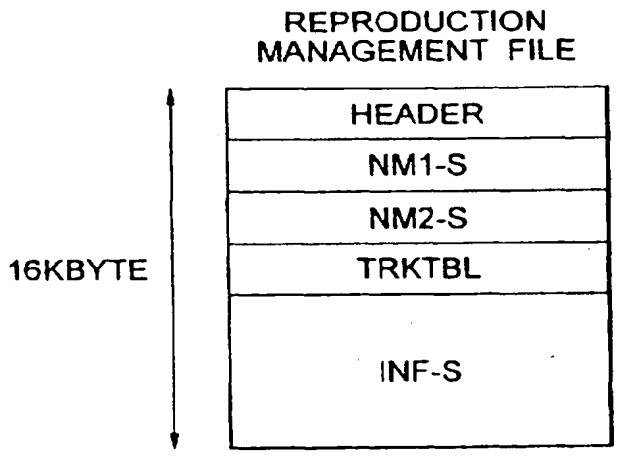
FIG. 4



STORAGE DATA OF FLASH MEMORY 34 OF PORTABLE STORAGE DEVICE 3

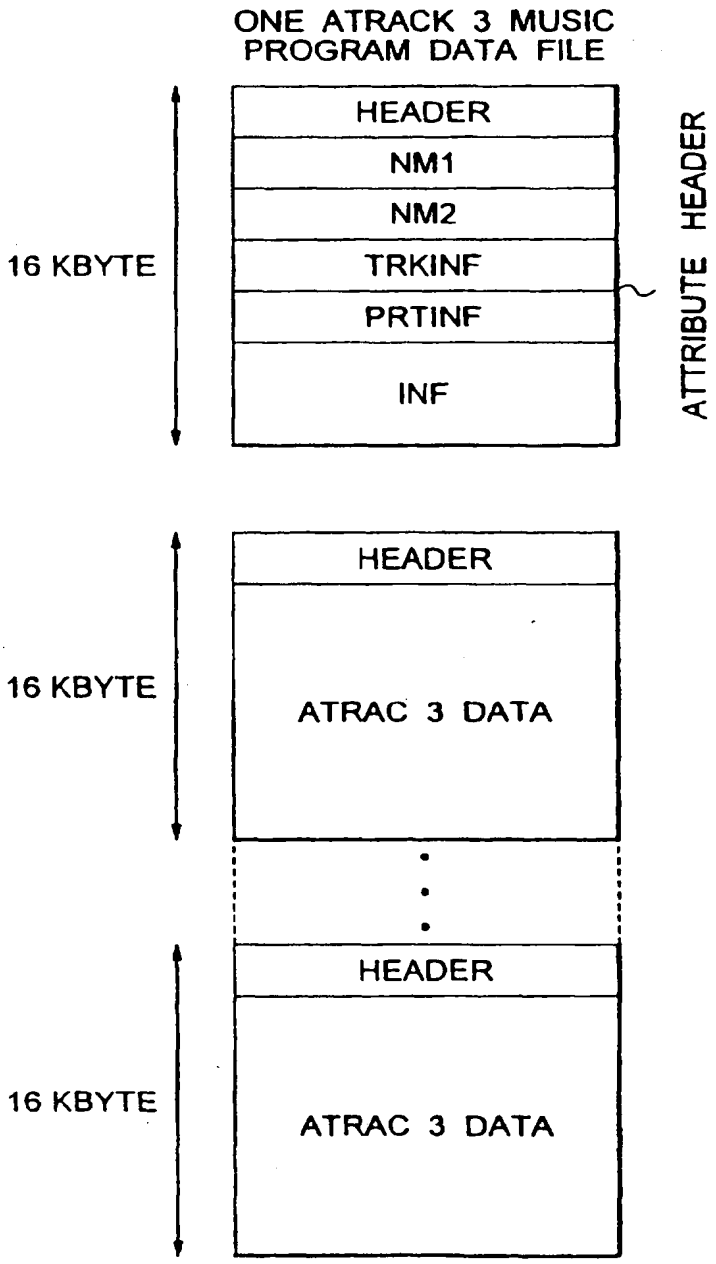
NSDOCID: <EP_____1037131A2_I_>

FIG. 5



3NSDOCID <EP 1037131A2 I >

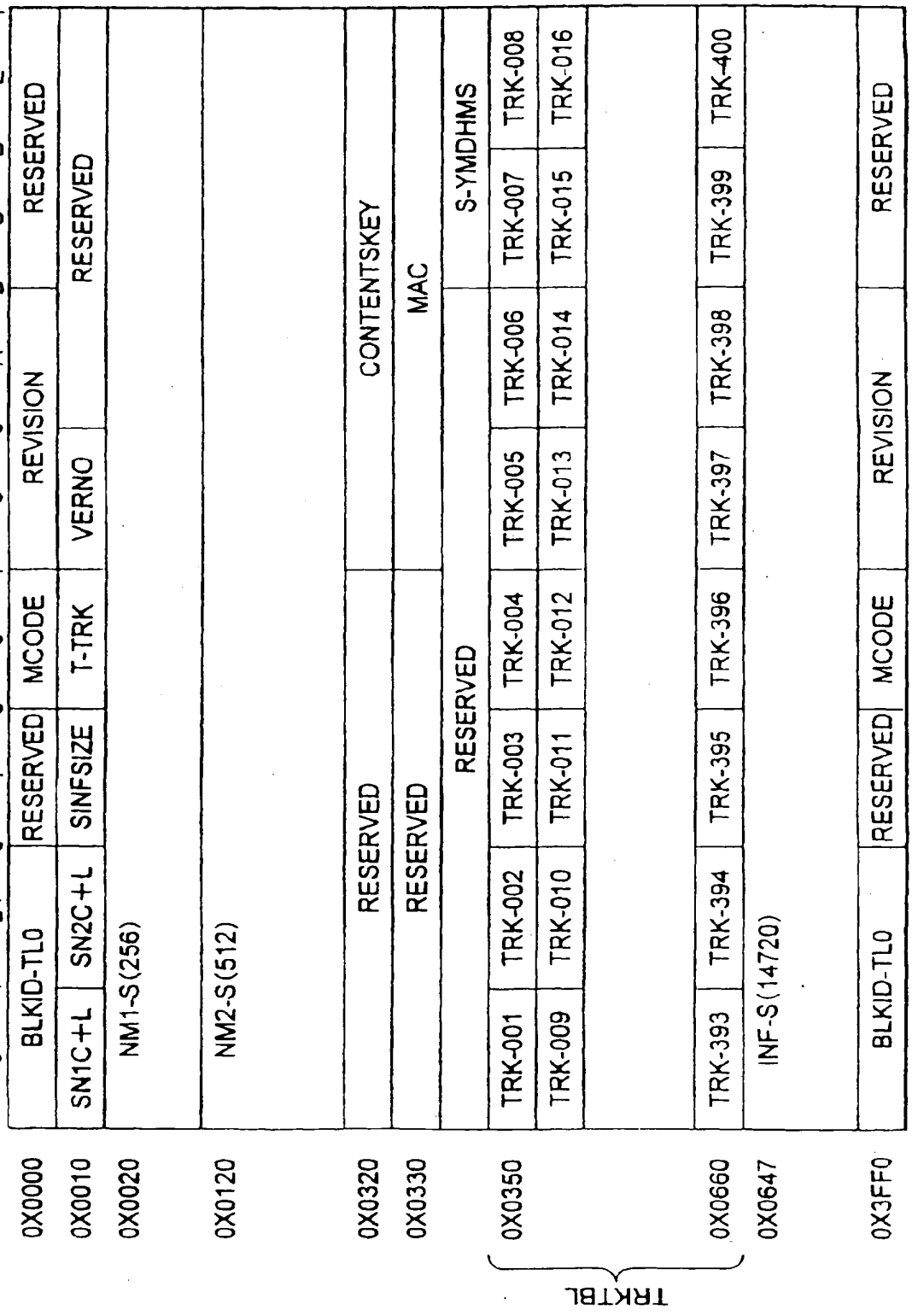
FIG. 6



SDOCID: <EP_____1037131A2_I_>

FIG. 7

REPRODUCTION MANAGEMENT FILE



3NSDOCID: <EP 1037131A2 I_>

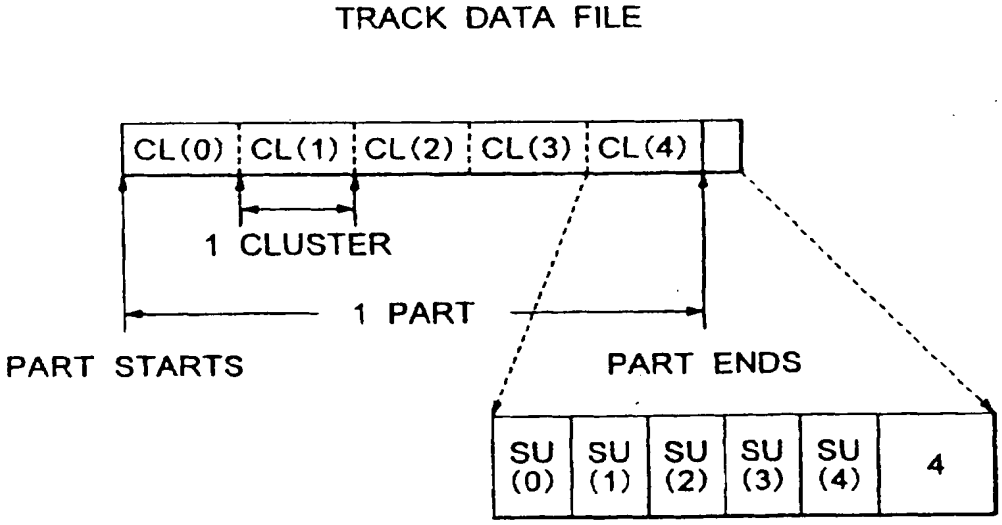
FIG. 8

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0X0000	BLKID-TL0 '				RESERVED		MCODE		REVISION				RESERVED			
0X0010	SN1C+L		SN2C+L		SINFSIZE		T-TRK		VERNO				RESERVED			
0X0020	NM1-S(256)															
0X0120	NM2-S(512)															
0X0320	RESERVED								CONTENTSKEY							
0X0330	RESERVED								MAC							
0X0350	RESERVED											S-YMDHMS				
0X0360	TRK-001		TRK-002		TRK-003		TRK-004		TRK-005		TRK-006		TRK-007		TRK-008	
	TRK-009		TRK-010		TRK-011		TRK-012		TRK-013		TRK-014		TRK-015		TRK-016	
0X0660	TRK-393		TRK-394		TRK-395		TRK-396		TRK-397		TRK-398		TRK-399		TRK-400	
0X0670	INF-S(14720)															
0X3FF0	BLKID-TL0				RESERVED		MCODE		REVISION				RESERVED			

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	INF	0X00	ID	0X00	SIZE		MCODE		C+L		RESERVED		DATA VARIABLE LENGTH			

ISDOCID: <EP_____1037131A2_I_>

FIG. 9



BNSDOCID: <EP_____1037131A2 I >

FIG. 10

A3Dnnnnn.MSA(ATRAC3 DATA FILE)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0X0000	BLKID−HD0				RESERVED		MCODE		RESERVED				BLOCK SERIAL			
0X0010	N1C+L		N2C+L		INFSIZE		T-PRT		T-SU				INX		XT	
0X0020	NM1(256)															
0X0120	NM2(512)															
0X0310																
0X0320	RESERVED(8)								CONTENTSKEY							
	RESERVED(8)								MAC							
	RESERVED(12)												A	LT	FNO	
	MG(D)SERIAL-NNN															
0X0360	CONNUM				YMDHMS-S				YMDHMS-E				MT	CT	CC	CN
0X0370	PRTSIZE				PRTKEY							RESERVED(8)				
0X0380					CONNUM0				PRTSIZE(0x0388)				PRTKEY			
0X0390					RESERVED(8)								CONNUM0			
	INF(0x0400)															
0X3FFF	BLKID−HD0				RESERVED		MCODE		RESERVED				BLOCK SERIAL			
0X4000	BLKID−A3D				RESERVED		MCODE		CONNUM0				BLOCK SERIAL			
0X4010	BLOCK SEED								INITILIZATION VECTOR							
0X4020	SU-000(NBYTE=384BYTE)															
0X41A0	SU-001(NBYTE)															
0X4320	SU-002(NBYTE)															
0X04A0	SU-041(NBYTE)															
0X7DA0																
0X7F20	RESERVED(NBYTE=208BYTE)															
	BLOCK SEED															
0X7FF0	BLKID−A3D				RESERVED		MCODE		CONNUM0				BLOCK SERIAL			

NSDOCID: <EP_____1037131A2_I_>

FIG. 11

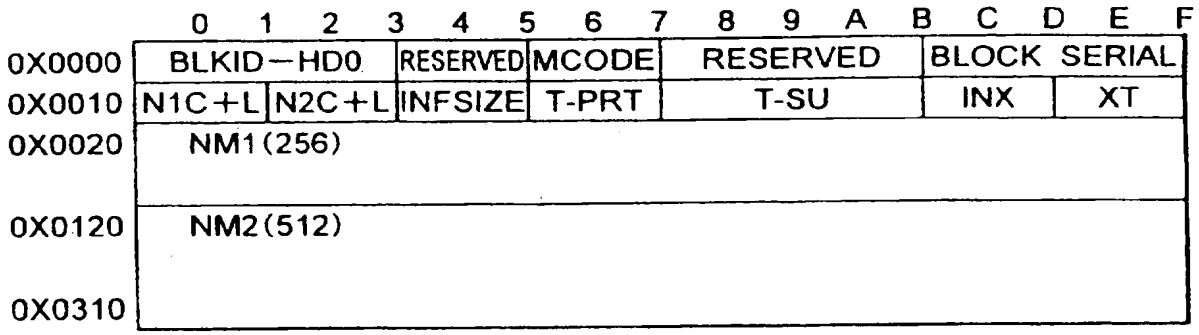
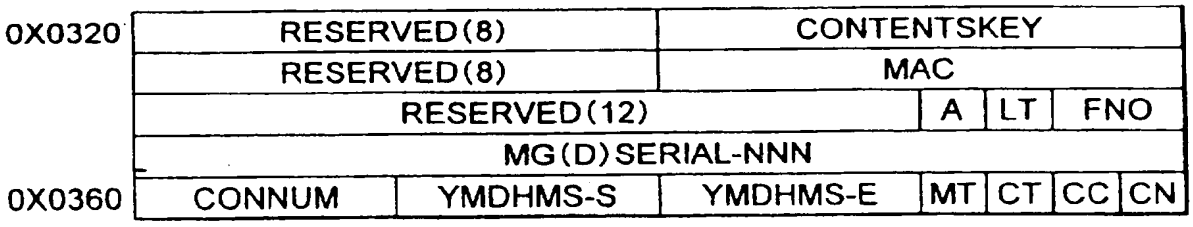


FIG. 12



3NSDOCID <EP 1037131A2 I >

FIG. 13

BIT7:MODE OF ATRAC3 0:DUAL 1:JOINT

BIT6.5.4 N OF 3 BITS:MODE VALUE

N	MODE	TIME	TRAMISSION RATE	SU	BYTES
7	HQ	47MIN	176KBPS	31SU	512
6		58MIN	146KBPS	38SU	424
5	EX	64MIN	132KBPS	42SU	384
4	SP	81MIN	105KBPS	53SU	304
3		90MIN	94KBPS	59SU	272
2	LP	128MIN	66KBPS	84SU	192
1	MONO	181MIN	47KBPS	119SU	136
0	MONO	258MIN	33KBPS	159SU	96

BIT3:RESERVED

BIT2:DATA TYPE 0:AUDIO 1:OTHER

BIT1:REPRODUCTION SKIP 0:NORMAL REP 1:SKIP

BIT0:EMPHASIS 0:OFF 1:ON(50／15 μ S)

NSDOCID: <EP_____1037131A2_I_>

FIG. 14

BIT7	:COPY PERMISSION 0:COPY PROHIBITION 1:COPY PERMISSION
BIT6	:GENERATION 0:ORIGINAL 1:FIRST OR LATER COPY GENERATION
HCMS BIT5-4	:COPY CONTROL FOR HIGH SPEED DIGITAL COPY
	00:COPY PROHIBITION 01:COPY FIRST GENERATION 10:COPY PERMISSION
	COPY OF FIRST COPY GENERATION IS PROHIBITED
BIT3-2	:MAGICGATE AUTHENTICATION LEVEL
	00:LEVEL10(NON-MG) 01:LEVEL1
	10:LEVEL2 11:REVERVED
	DIVIDE AND COMBINE ARE PROHIBITED IN OTHER THAN LEVEL 10
BIT1.0	RESERVED

FIG. 15

0X0370	PRTSIZE	PRTKEY		RESERVED(8)
0X0380		CONNUM0	PRTSIZE(0x0388)	PRTKEY
0X0390		RESERVED(8)		CONNUM0

FIG. 16

0X4000	BLKID−A3D	RESERVED	MCODE	CONNUM0	BLOCK SERIAL
0X4010	BLOCK SEED			INITILIZATION VECTOR	
0X4020	SU-000(NBYTE=384BYTE)				

3NSDOCID: <EP ..... 1037131A2_I_>

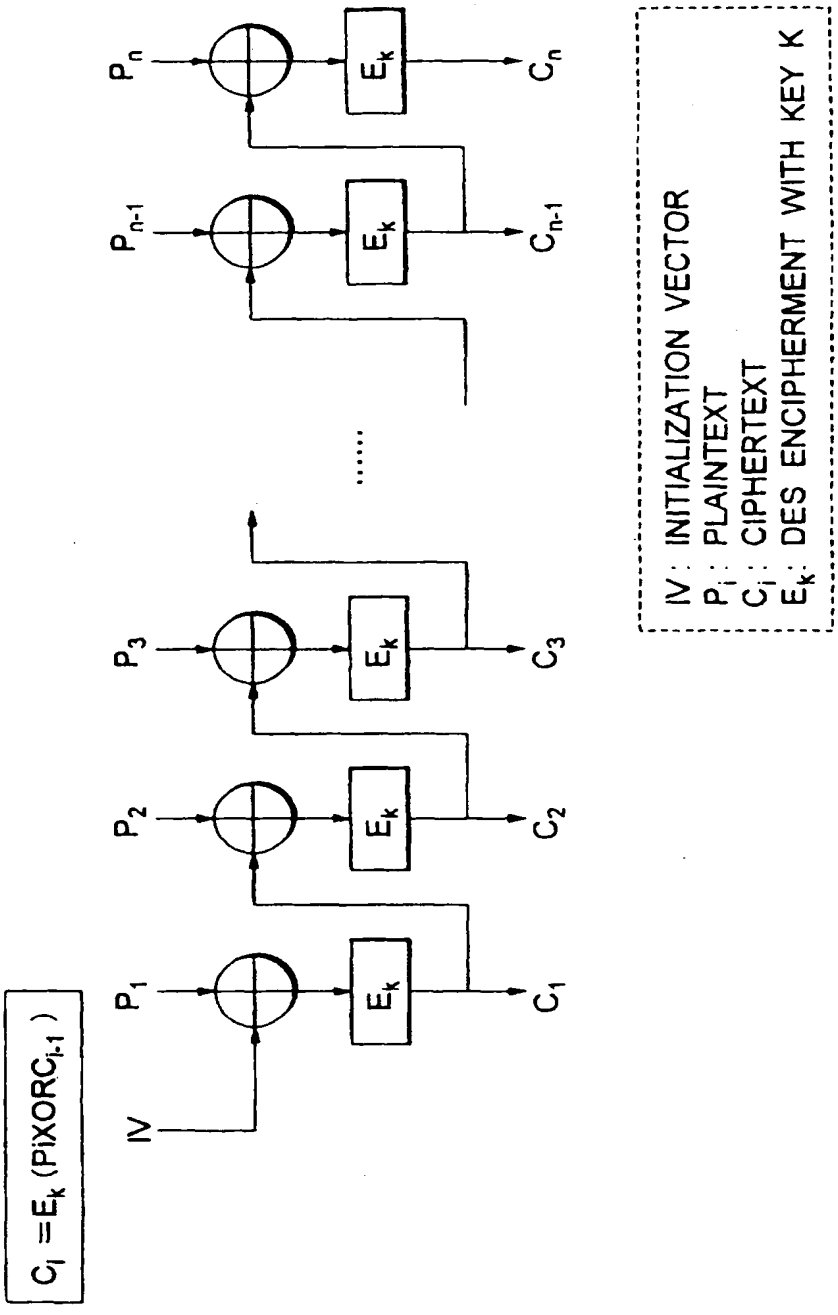
FIG. 17

DATA TO BE STOREND IN STORAGE UNIT 61 OF PORTABLE PLAYER 4

MASTER KEY DATA	MK_0
	MK_1
	MK_2
	MK_3
	⋮
	IK_{30}
	IK_{31}
DEVICE IDENTIFICATION DATA	ID_d

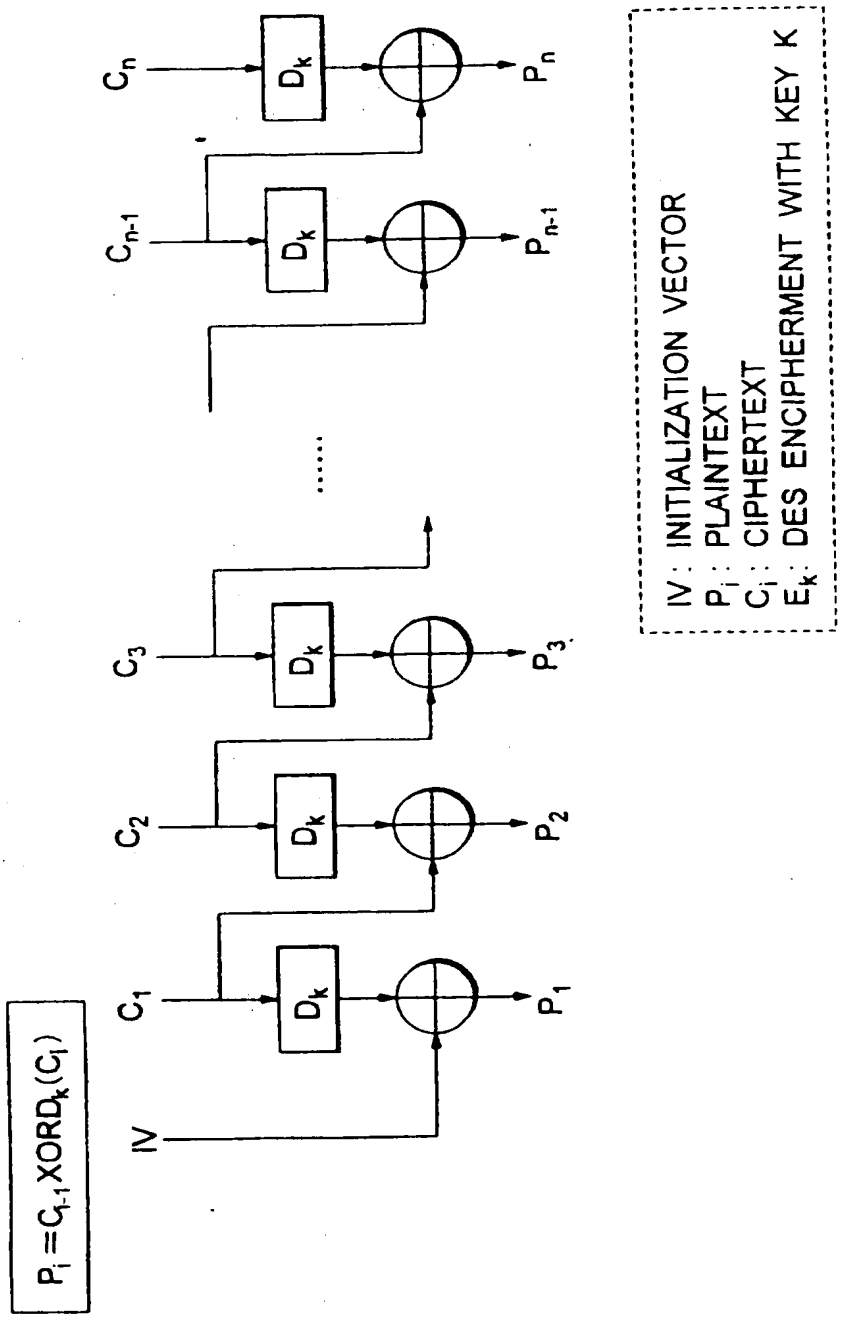
NSDOCID: <EP_____1037131A2_I_>

FIG. 18



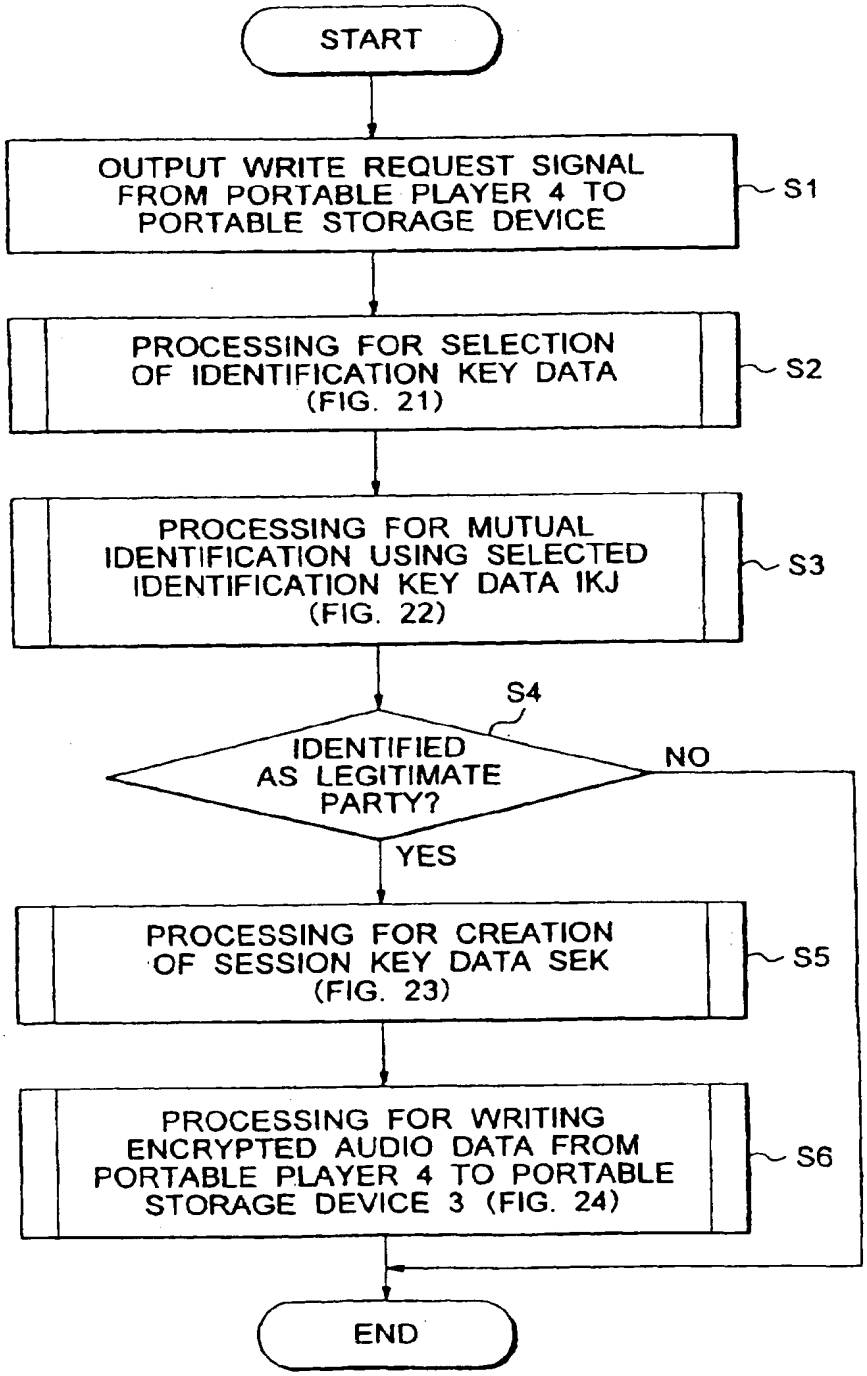
3NSDOCID: <EP 1037131A2 I >

FIG. 19



ISDOCID: <EP_____1037131A2_I_>

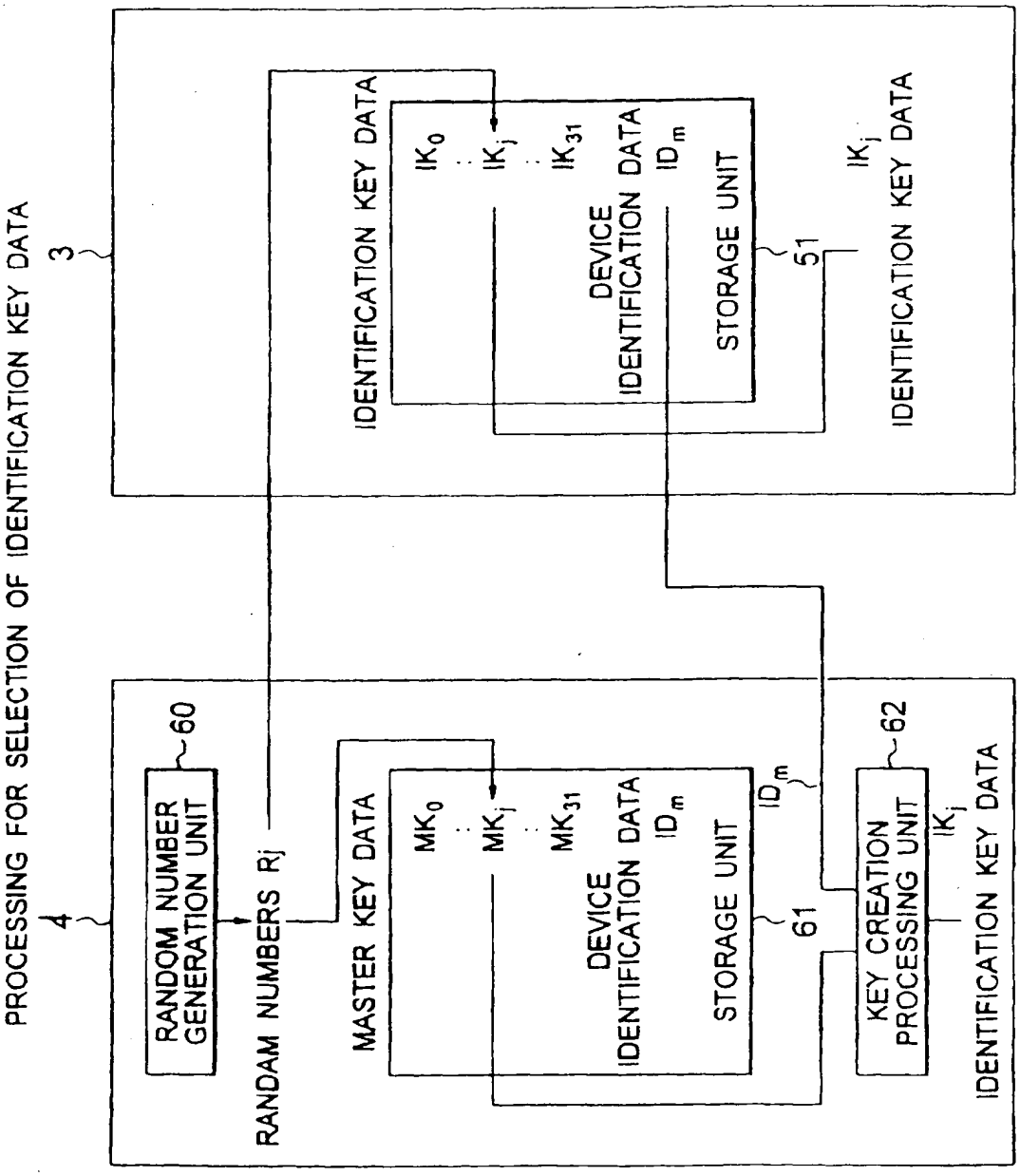
FIG. 20



WRITE PROCESSING TO PORTABLE STORAGE DEVICE 3

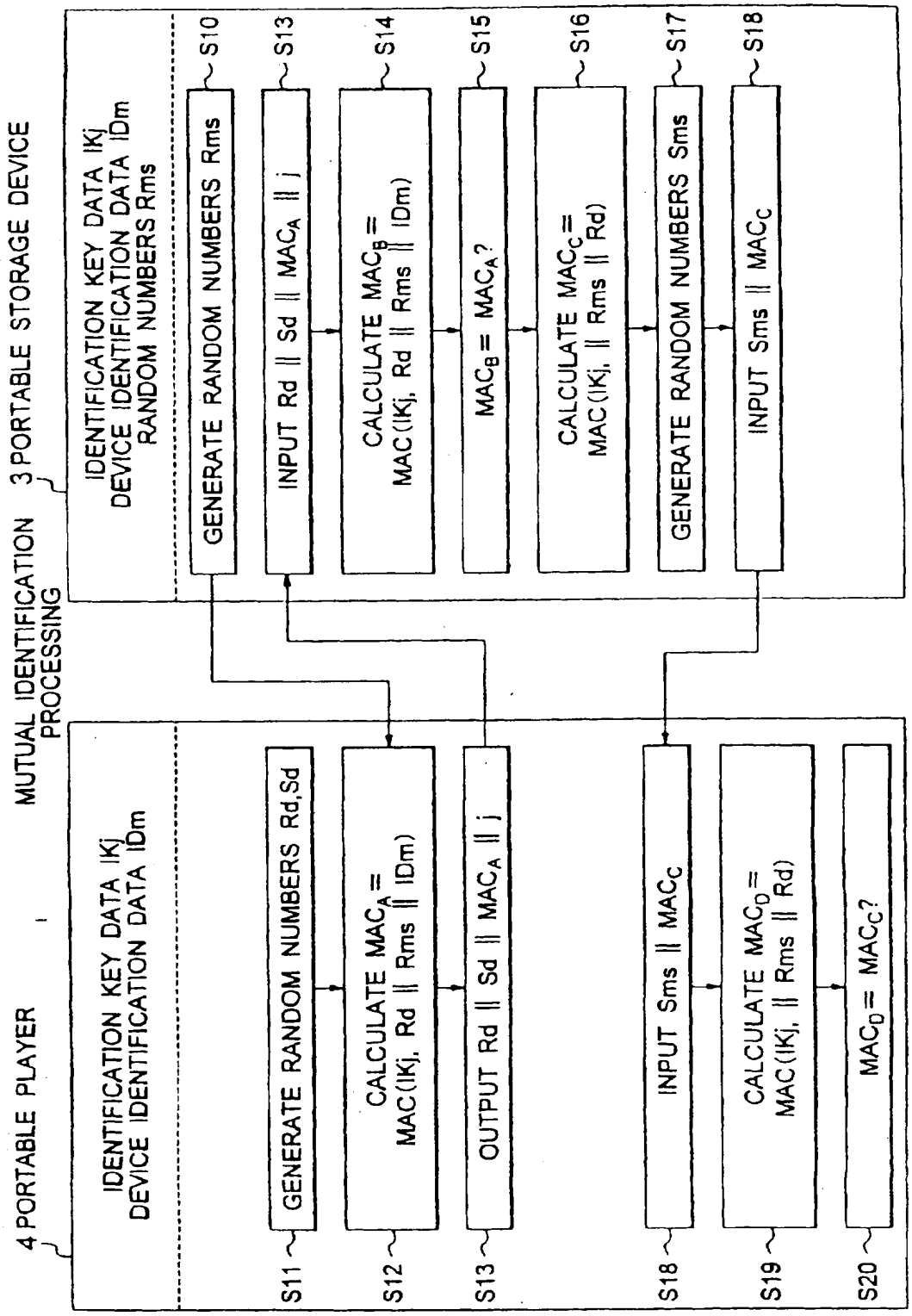
3NSDOCID: <EP____ ___1037131A2_I_>

FIG. 21



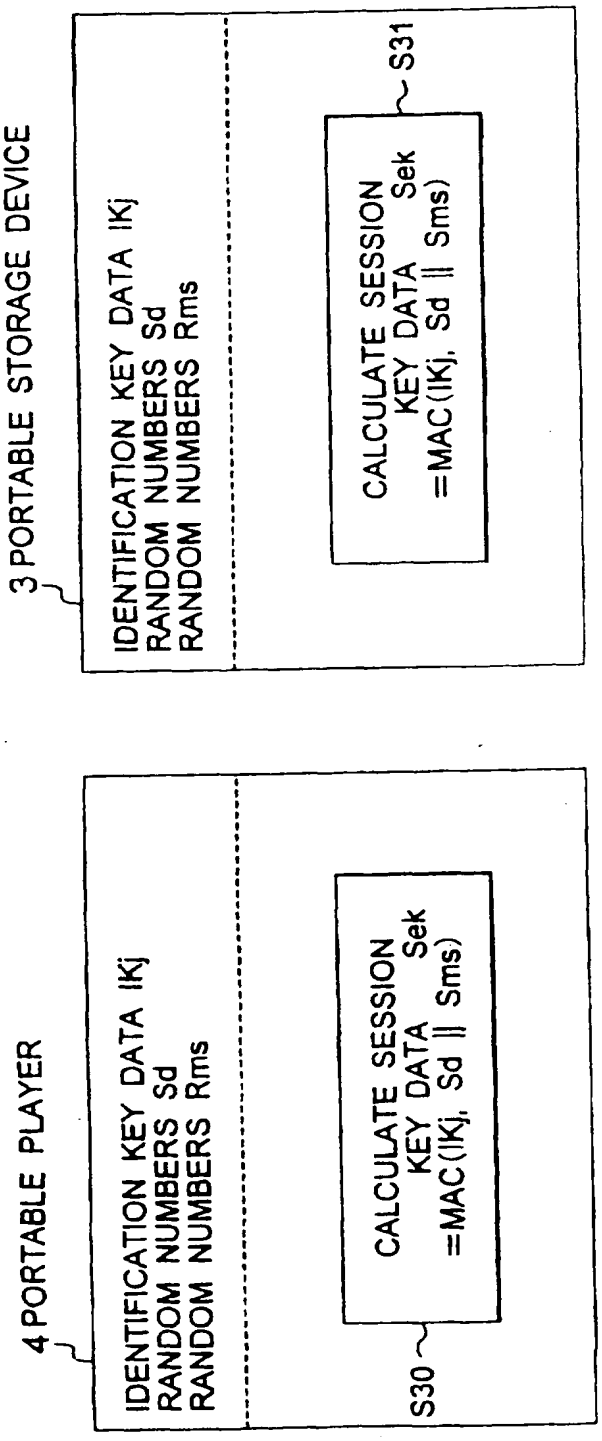
NSDOCID: <EP..._____1037131A2_I_>

FIG. 22



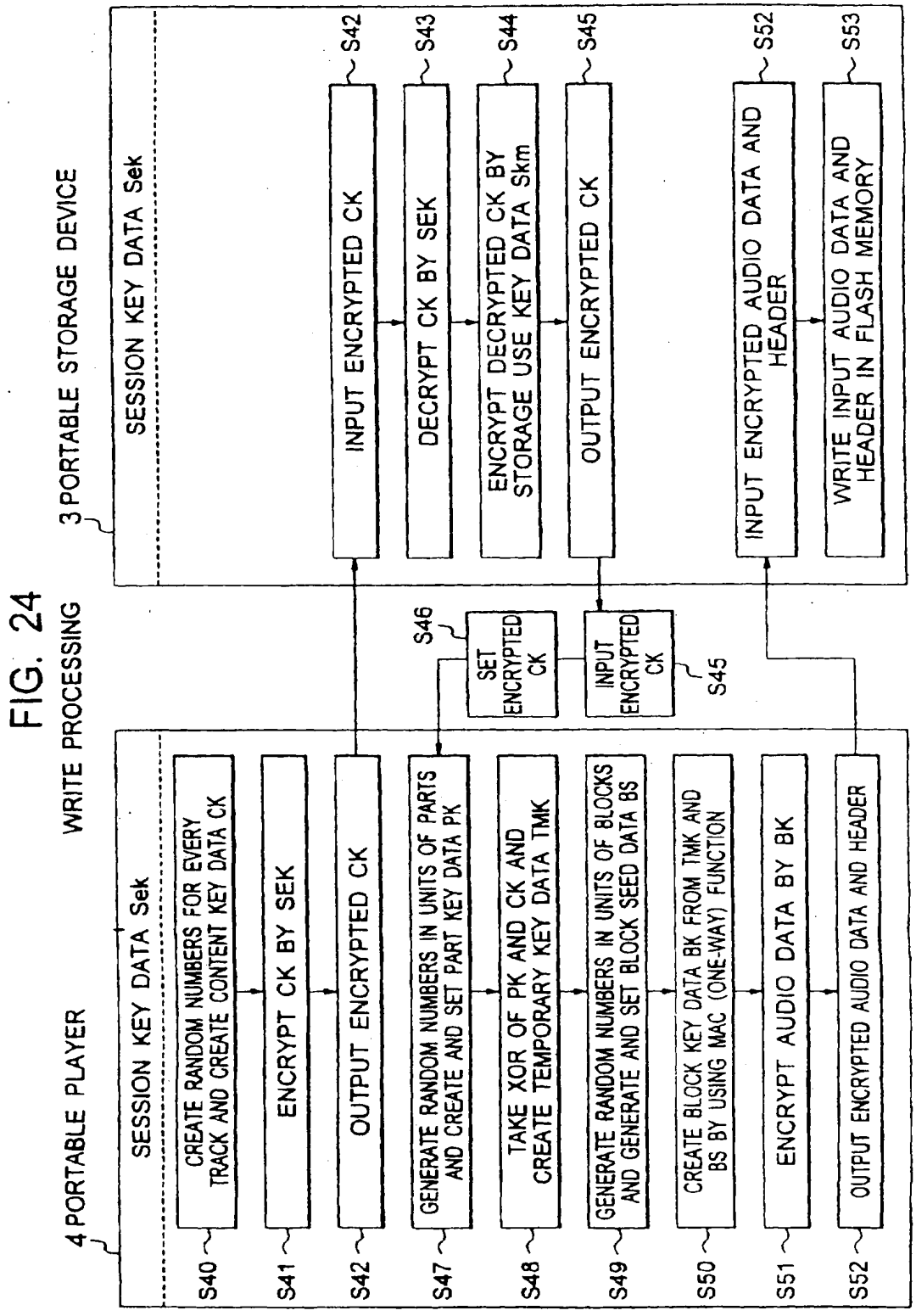
EPNSDOCID: <EP_____1037131A2 I_>

FIG. 23



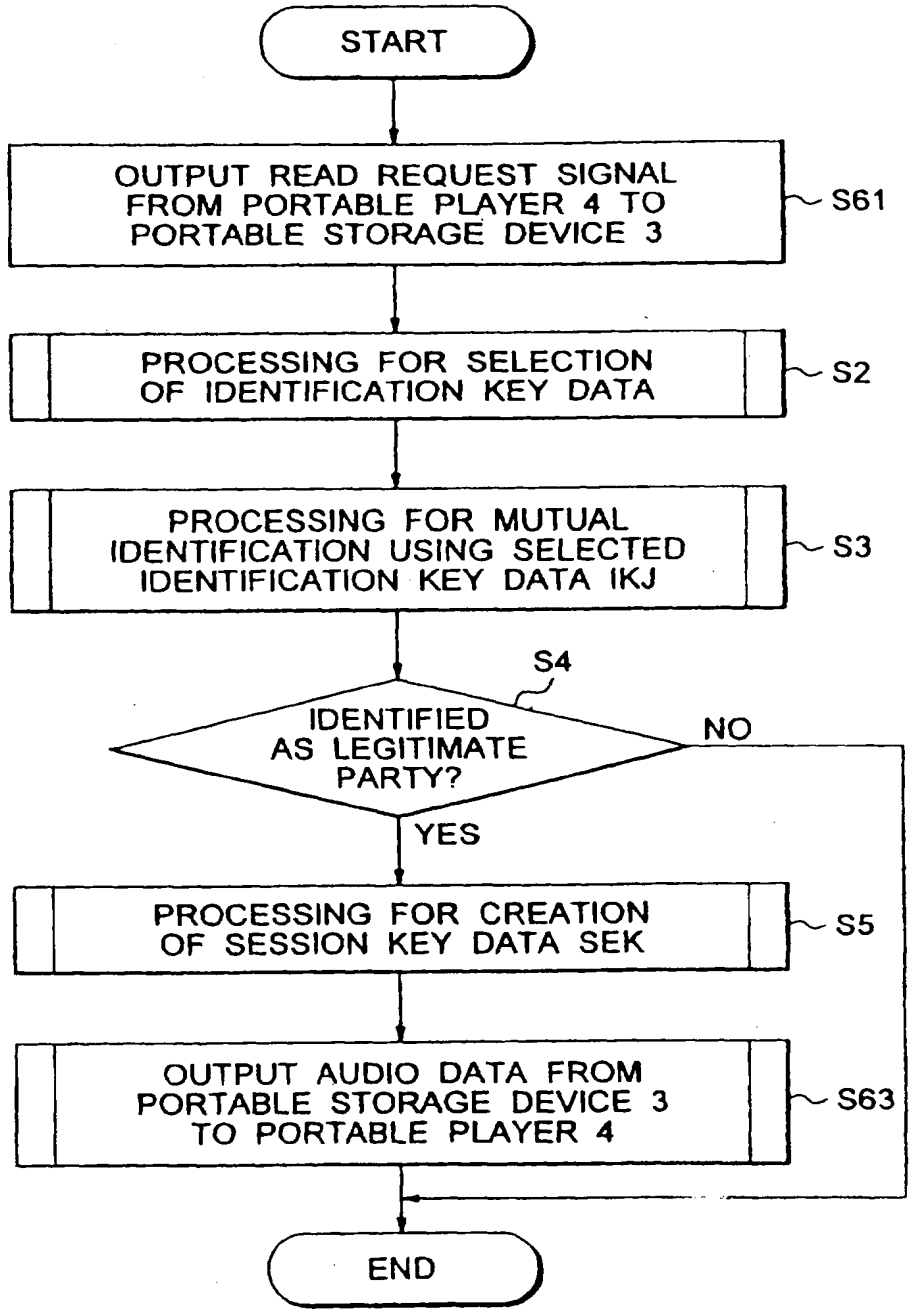
NSDOCID: <EP_____1037131A2_I_>

FIG. 24



3NSDOCID: <EP_____1037131A2_I_>

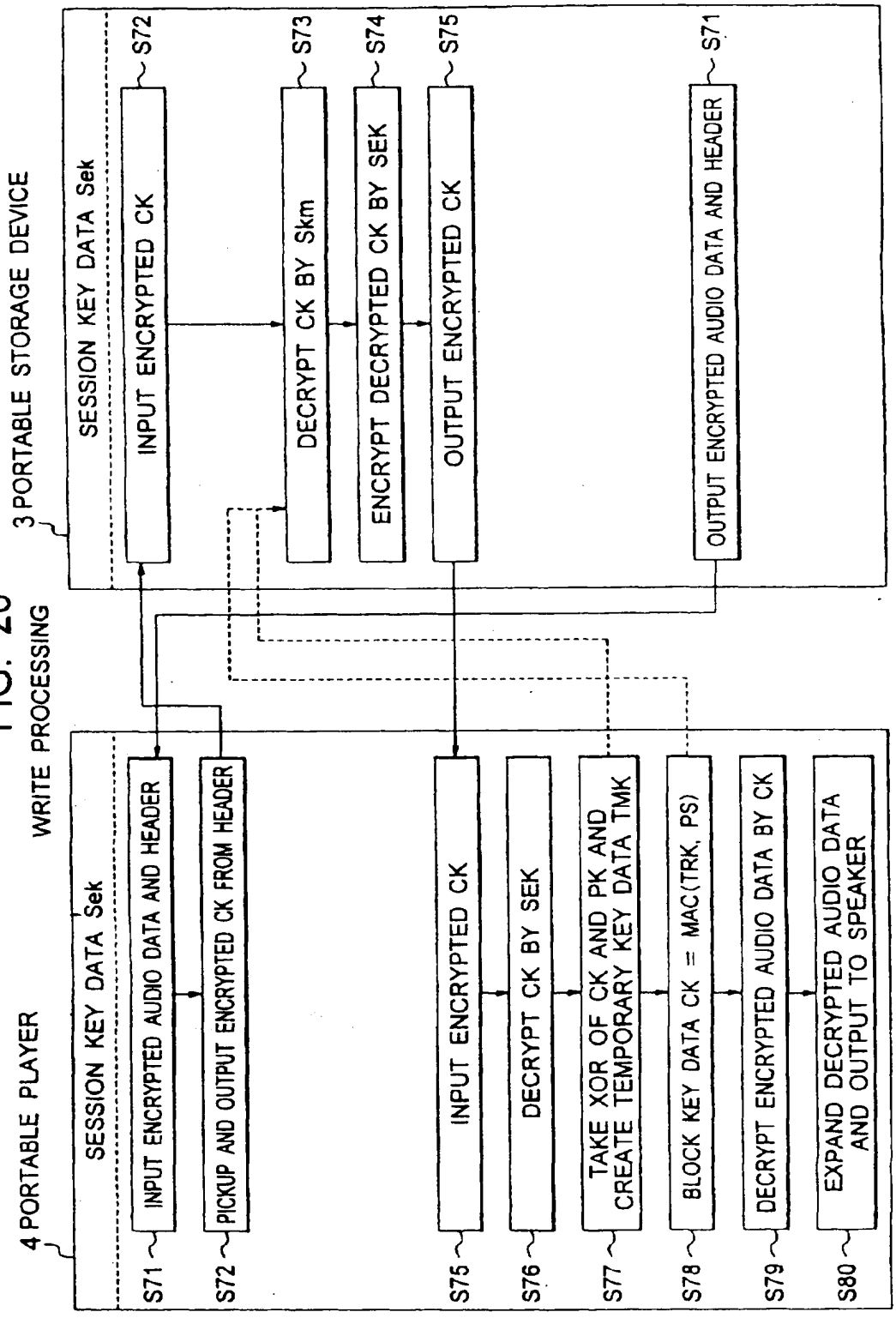
FIG. 25



PROCESSING FOR READING FROM PORTABLE STORAGE DEVICE 3

NSDOCID: <EP_____1037131A2 I. >

FIG. 26



3NSDOCID: <EP_____ ___ 1037131A2 I >

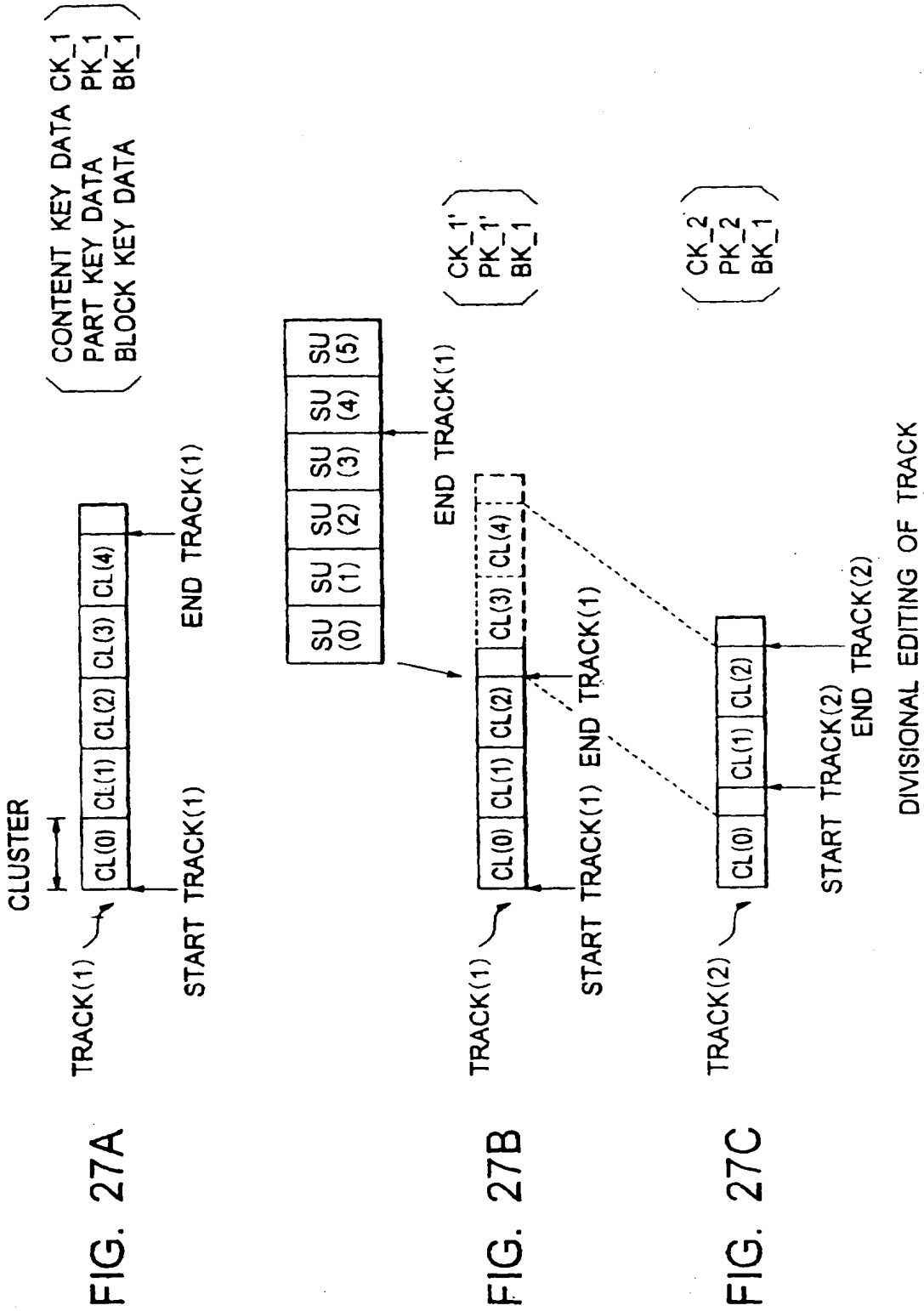


FIG. 27A

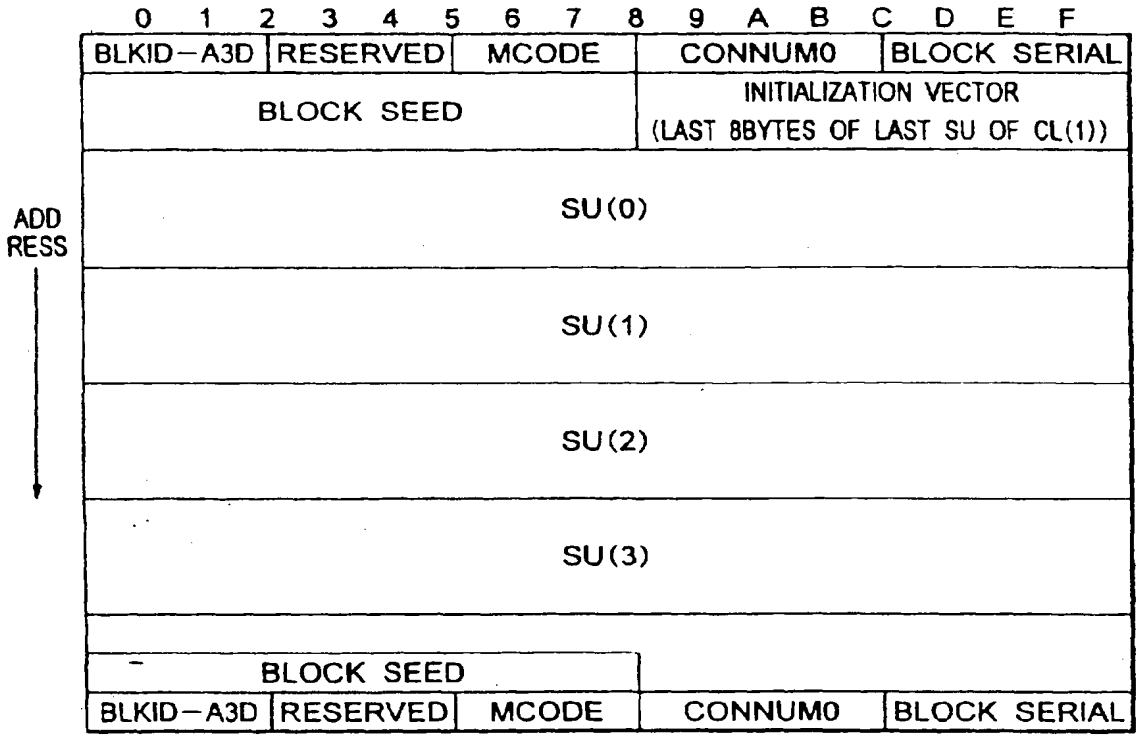
FIG. 27B

FIG. 27C

JSDOCID: <EP_____1037131A2_I_>

FIG. 28

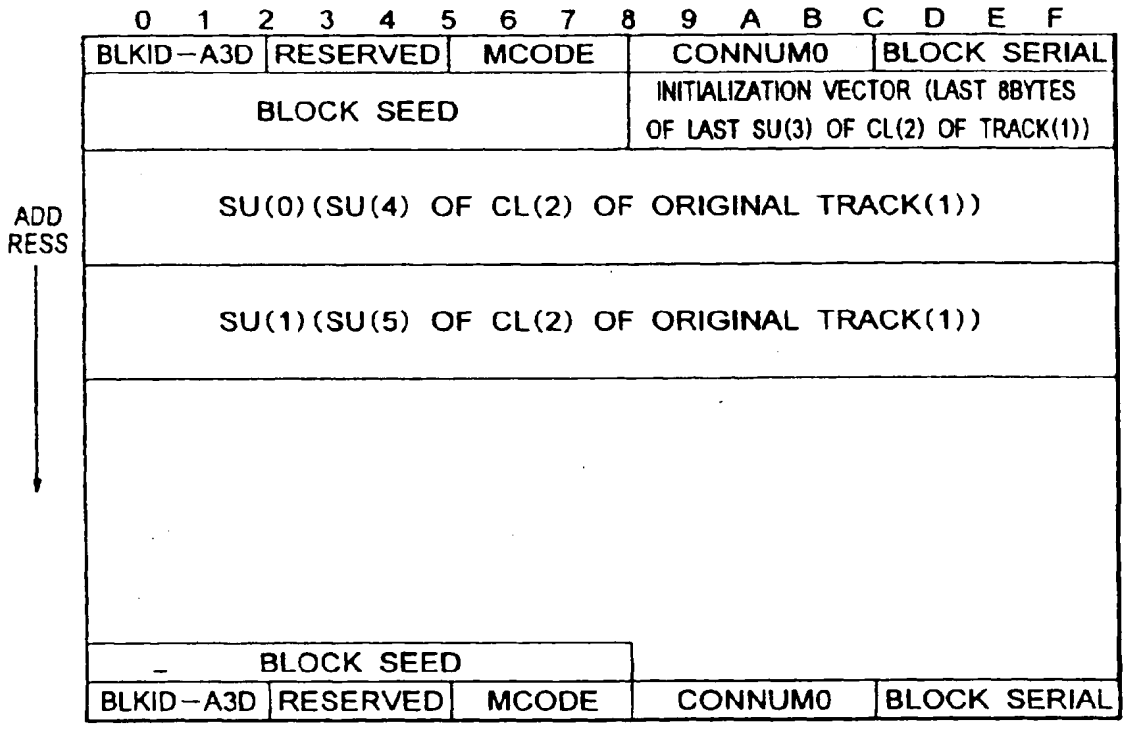
CLUSTER CL(2) OF TRACK(1) AFTER DIVIDE



3NSDOCID: <EP_ _____1037131A2_I_>

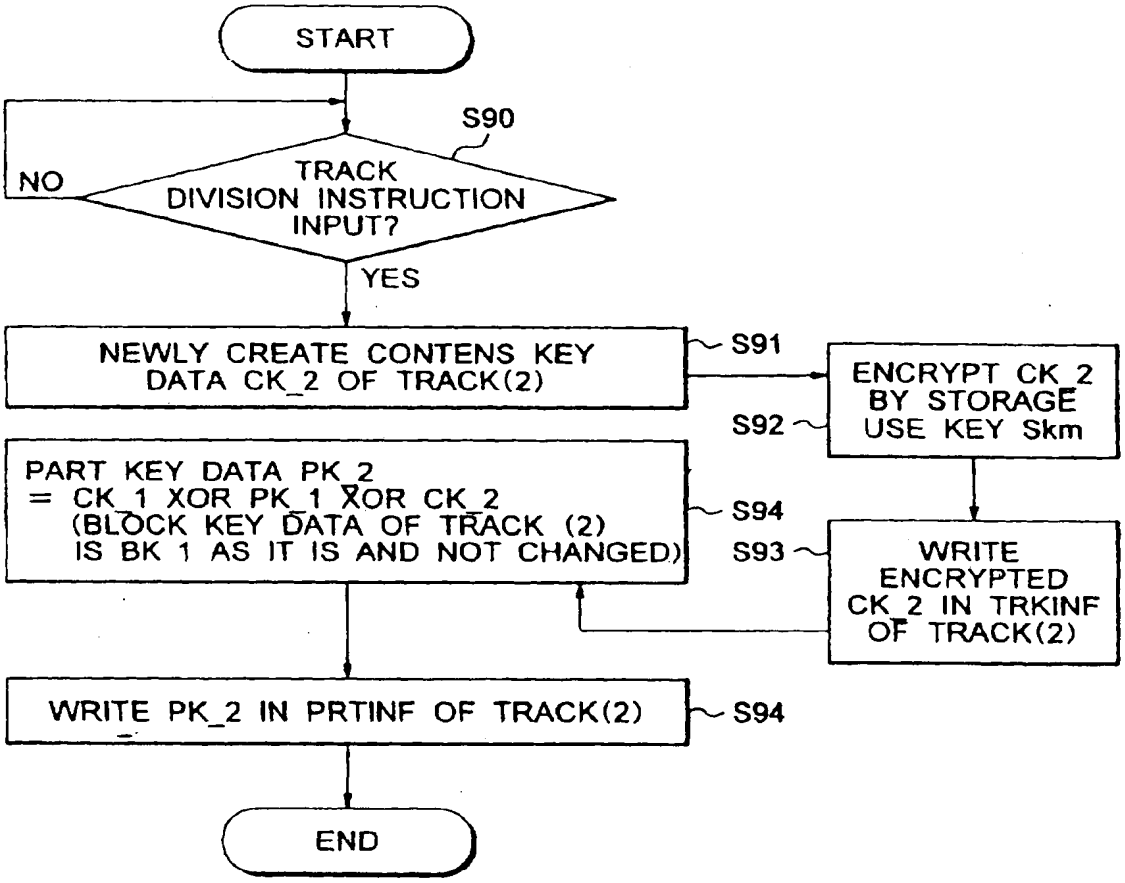
FIG. 29

CLUSTER CL(0) OF TRACK(2) AFTER DIVIDE



NSDOCID: <EP_____1037131A2_I_>

FIG. 30



3NSDOCID: <EP_ _ _ _ 1037131A2_I_ >

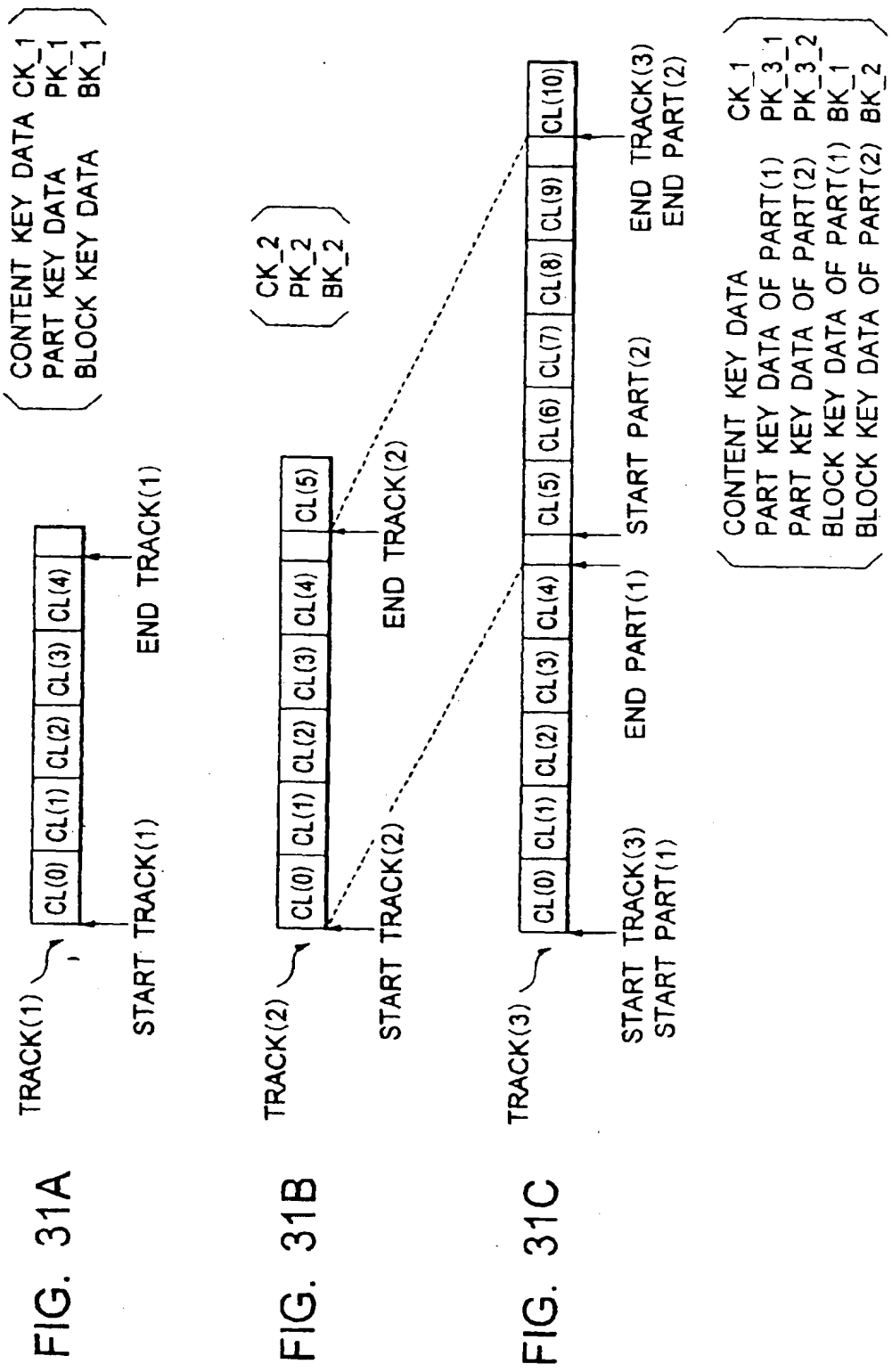


FIG. 31A

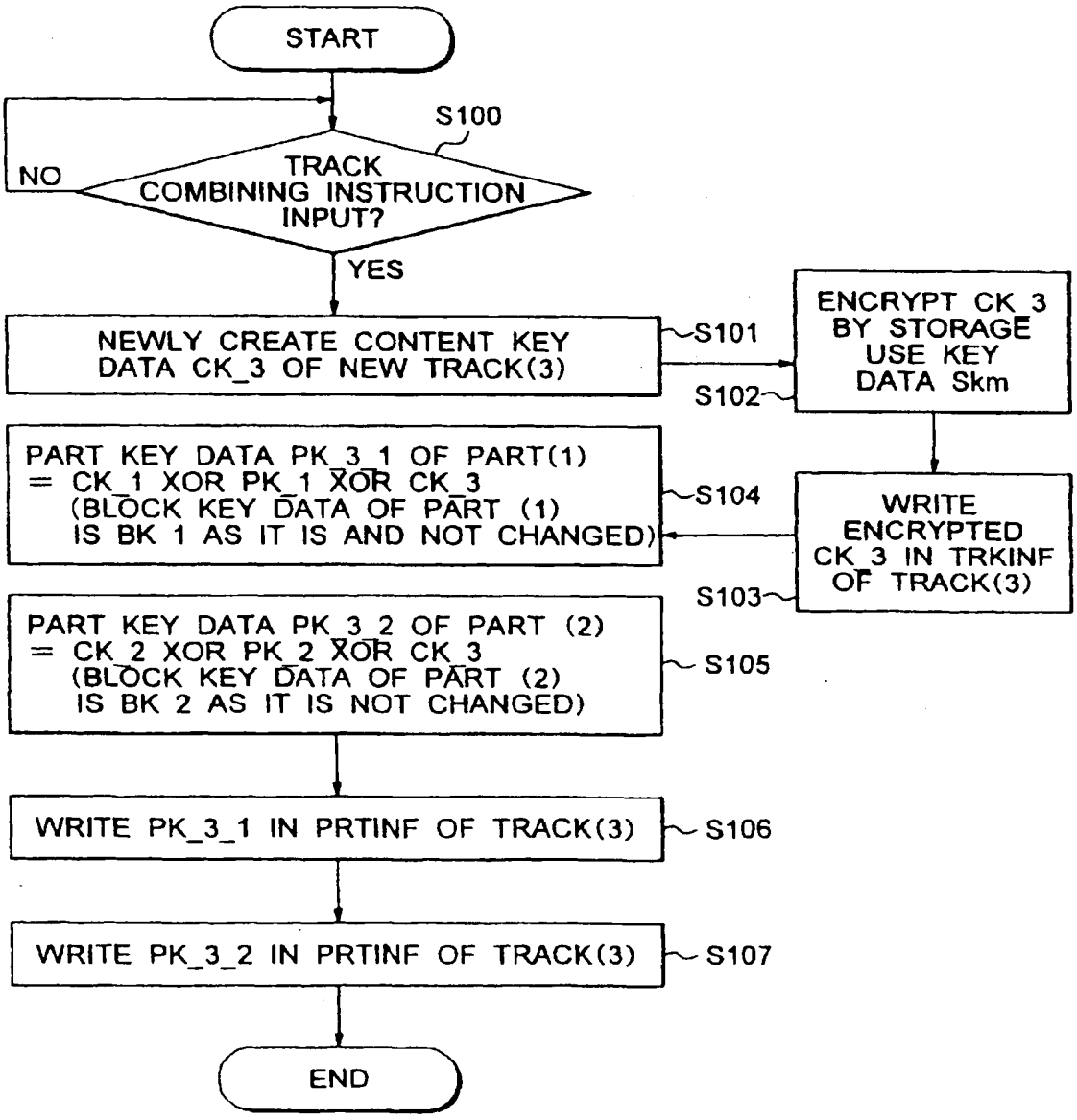
FIG. 31B

FIG. 31C

TRACK COUPLING PROCESSING

↓SDOCID: <EP_____1037131A2_I_>

FIG. 32



3NSDOCID <EP_____1037131A2_I_>

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 037 131 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
14.09.2005 Bulletin 2005/37

(43) Date of publication A2:
20.09.2000 Bulletin 2000/38

(21) Application number: **00302122.7**

(22) Date of filing: **15.03.2000**

(51) Int Cl.[7]: **G06F 1/00**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **15.03.1999 JP 6915199**
24.06.1999 JP 17818899

(71) Applicant: **SONY CORPORATION**
Tokyo 141 (JP)

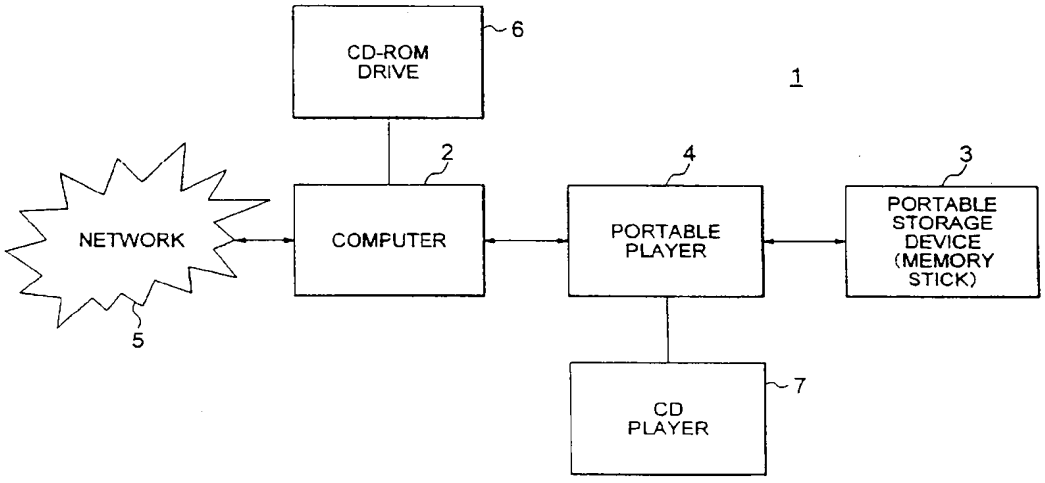
(72) Inventors:
- **Oishi, Tateo, Sony Corporation**
Tokyo (JP)
- **Asano, Tomoyuki, Sony Corporation**
Tokyo (JP)
- **Kihara, Nobuyuki, Sony Corporation**
Tokyo (JP)
- **Yokota, Teppei, Sony Corporation**
Tokyo (JP)

(74) Representative: **Turner, James Arthur et al**
D Young & Co
120 Holborn
London EC1N 2DY (GB)

(54) Data processing system and method for mutual identification between apparatuses

(57) A storage unit in a first mutual identification unit stores master key data and a second storage unit in a second mutual identification unit stores identification key data. A random number generating unit generates a number that is used to select a master key and a corresponding identification key at the mutual identification units. The first mutual identification unit generates an estimate of the selected identification key using the selected master key and uses this estimate as a common key when performing mutual identification with the second mutual identification unit.

FIG. 1



EP 1 037 131 A3

BNSDOCID: <EP 1037131A3 I >

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 30 2122

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	LAMPSON B ET AL: "AUTHENTICATION IN DISTRIBUTED SYSTEMS: THEORY AND PRACTICE" OPERATING SYSTEMS REVIEW (SIGOPS), ACM HEADQUARTER. NEW YORK, US, vol. 25, no. 5, January 1991 (1991-01), pages 165-182, XP000288310 * page 170, left-hand column *	1,2, 7-11, 16-18	G06F1/00
A	WOO T Y C ET AL: "AUTHENTICATION FOR DISTRIBUTED SYSTEMS" COMPUTER, IEEE COMPUTER SOCIETY, LONG BEACH., CA, US, US, vol. 25, no. 1, January 1992 (1992-01), pages 39-52, XP000287833 ISSN: 0018-9162 * the whole document *	1-18	
A	HARN L ET AL: "ID-BASED CRYPTOGRAPHIC SCHEMES FOR USER IDENTIFICATION, DIGITAL SIGNATURE, AND DEY DISTRIBUTION" 1 June 1993 (1993-06-01), IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE INC. NEW YORK, US, PAGE(S) 757-760 , XP000399843 ISSN: 0733-8716 Section V	1-18	TECHNICAL FIELDS SEARCHED (Int.Cl.7) G06F H04L
A	LIM CHAE HOON: "Message Encryption and Authentication Using One-Way Hash Functions" 1996, , XP002310898 * the whole document *	1-18	
A	JOYE M ET AL: "ID-BASED SECRET-KEY CRYPTOGRAPHY" October 1998 (1998-10), OPERATING SYSTEMS REVIEW (SIGOPS), ACM HEADQUARTER. NEW YORK, US, PAGE(S) 33-39 , XP000849893 * the whole document *	1-18	

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
The Hague	27 July 2005	Fleckinger, C

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

NSDOCID: <EP_____1037131A3_I_>